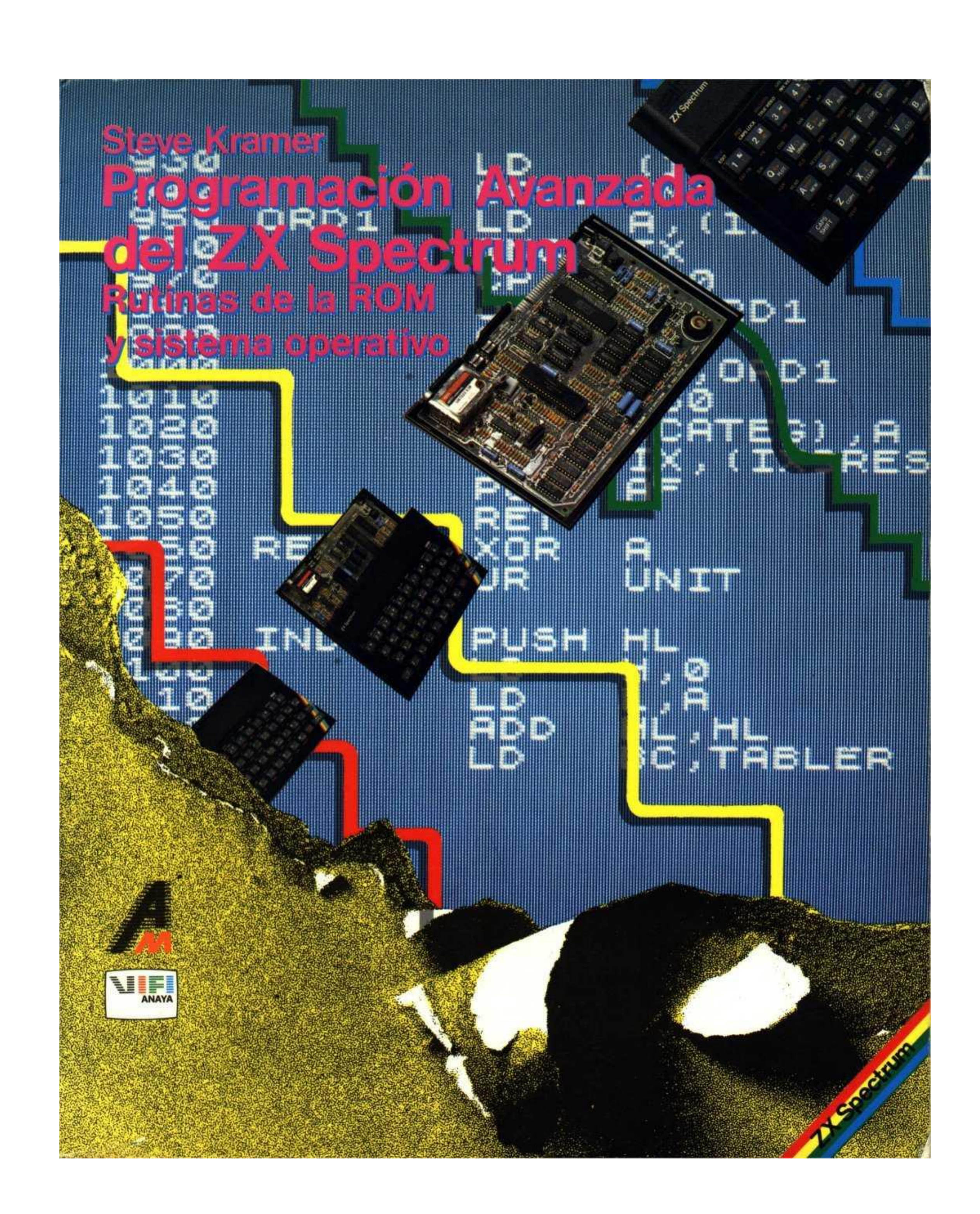


Steve Kramer
Programación Avanzada del ZX Spectrum
Rutinas de la ROM y sistema operativo
ANAYA
ZX Spectrum

Programación avanzada del ZX Spectrum

Rutinas de la ROM
y Sistema Operativo

Steve Kramer



ANAYA MULTIMEDIA

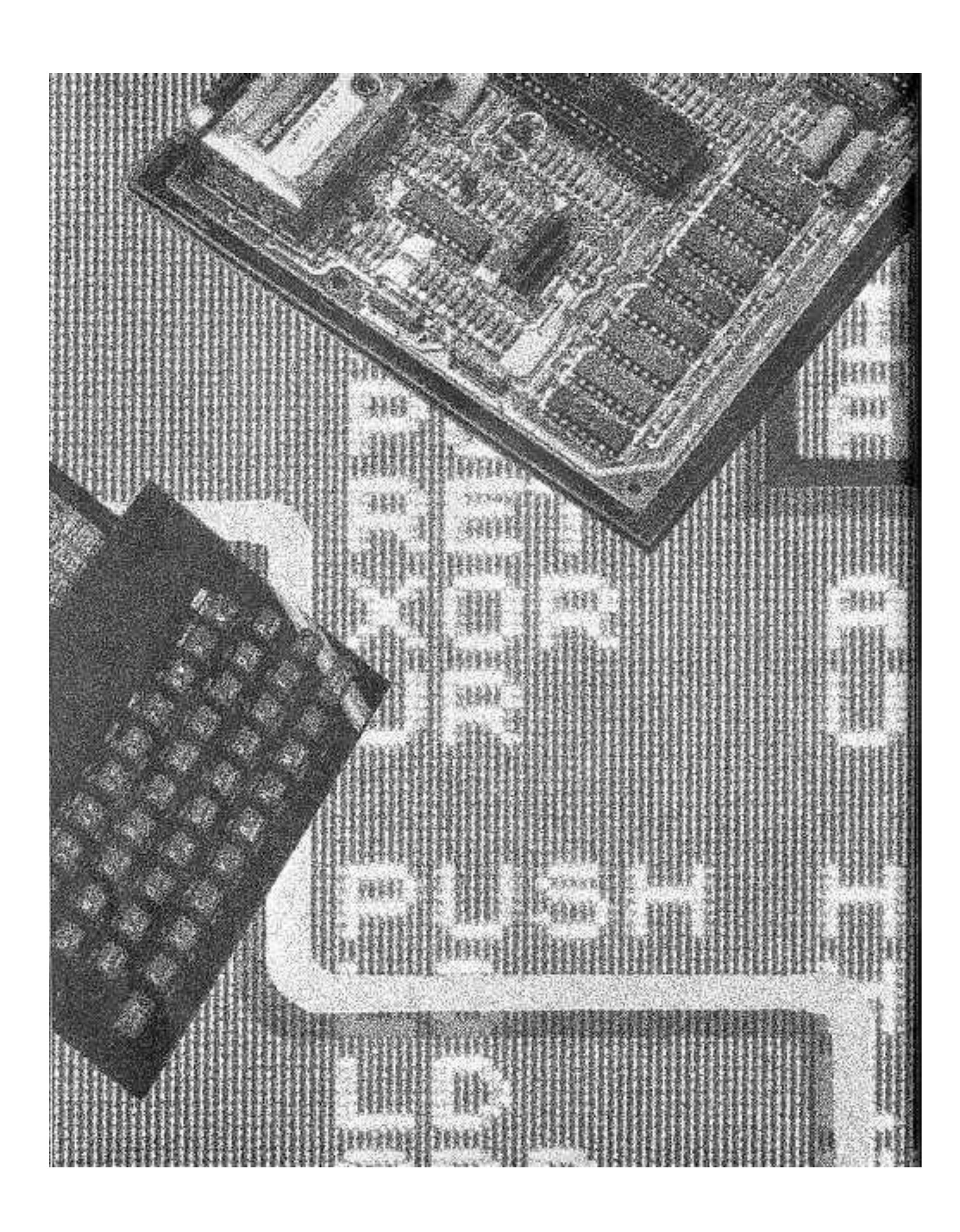
MICROINFORMATICA
Título de la obra original :
THE SPECTRUM OPERATING SYSTEM

Traducción de: Fernando García
Diseño de colección: Antonio Lax
Diseño de cubierta: Narcís Fernández

Primera edición, 1984
Micro Press
Castle House, 27 London Road
Tunbridge Wells, Kent

Villafranca, 22. 28028 Madrid
Depósito legal: M. 23.786-1985
ISBN: 84-7614-029-0
Imprime: Anzos, S. A. Fuenlabrada (Madrid)

Índice


PUSH

1. Introducción

Aunque este libro está dirigido, en principio, a aquellas personas que tienen ya algún conocimiento de la programación en lenguaje ensamblador, hay también una gran cantidad de información que podrá ser usada por los programadores inexpertos, a los que les gustaría poder tener acceso a la versatilidad del código máquina, aun sin conocerlo. Si usted pertenece al último grupo, yo espero que este libro le abra el apetito, y le sugiero que comience a aprender un poquito sobre la programación en lenguaje ensamblador. Con unos mínimos conocimientos y usando las rutinas de los capítulos siguientes se pueden escribir y usar fácilmente rutinas pequeñas en código máquina.

Para aquellas personas no familiarizadas con el sistema operativo del Spectrum o que prefieran, como yo, seguir el sistema cómodo usando las rutinas de la ROM y otras facilidades existentes --en vez de reinventar la rueda cada vez que quieren escribir un programa- discutiré cómo usar las llamadas a la ROM en las dos, en la de 16K en el Spectrum y en la de 8K en el interfaz del microdrive.

En algunos casos, daré ejemplos y descripciones de cómo usarlos, excepto donde sean una simple llamada (CALL).

Examinaré también las variables del sistema, enseñaré cómo usarlas en nuestro eneficio y explicaré el uso de las rutinas basadas en interrupciones; que permiten que los sprites puedan ser usados en el Spectrum estándar.

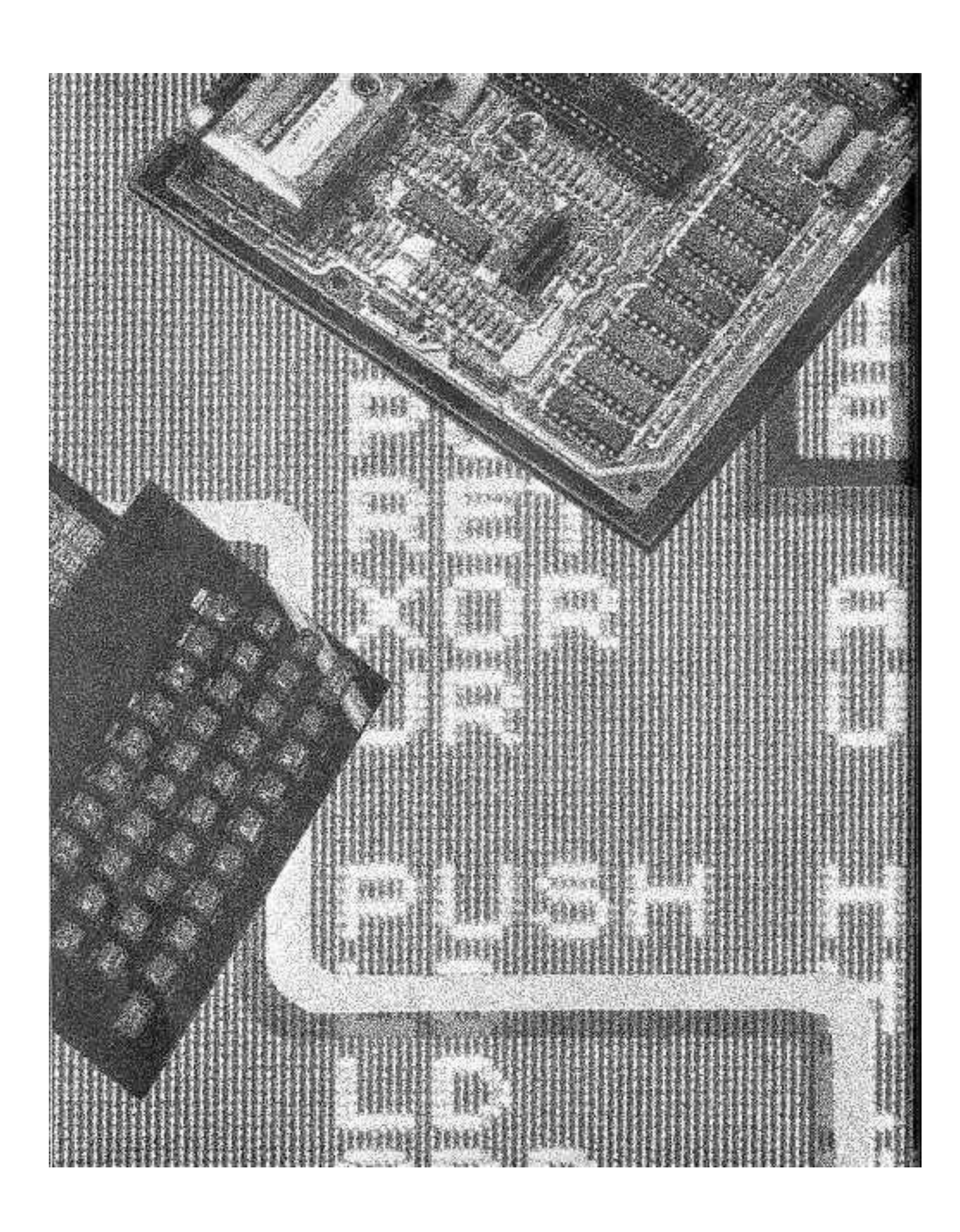
A no ser que usted desee convertir a mano las rutinas en ensamblador en números y los introduzca en la memoria de su ordenador a base de pokes existe un requisito necesario para ser capaz de usar este libro, es un programa ensamblador. Yo puedo recomendarle el Picturesque Editor Assembler y también sus programas monitor/desensamblador; son fáciles y rápidos de usar. También le puedo recomendar el Hisolft Devpack 3, que probablemente habrá sido mejorado cuando usted lea este libro. Para los usuarios más experimentados, el Devpack (que contiene el monitor/desensamblador y el editor/ensamblador) es, en mi opinión, el mejor hasta ahora. Las últimas versiones de ambos programas son compatibles con el Microdrive.

No es mi intención enseñar a programar en lenguaje ensamblador, cuando hay gran cantidad de libros sobre este tema que lo cubren con detalle. Puedo recomendarle el siguiente, Programación del Z80, escrito por Rodnay Zaks y editado por Anaya Multimedia. El primero no está dirigido al Spectrum, pero va más lejos en detalles que el segundo y también tiene descripciones concisas de todos los códigos de operaciones disponibles.

Para llamar a un programa en código máquina desde el BASIC usted puede usar una instrucción RANDOMIZE USR NN, un PRINT USR NN o un LET V = USR NN, donde NN es el punto de entrada en el programa de código máquina y V puede ser una variable numérica. A la vuelta de la rutina del código máquina, la variable en la instrucción LET será igual que el contenido del par de registros BC al salir de la rutina, y con la instrucción PRINT el valor de estos registros será impreso en la corriente actual. En todos los casos, el par de registros BC contendrá NN en la entrada a la rutina en código máquina.

Siempre que un programa en código máquina es llamado, es conveniente salvar el contenido del par de registros H'L', pues el contenido de éstos será necesario para lograr que el control se devuelva al programa BASIC. El par de registros IY no deberá ser usado normalmente debido a que la ROM lo usa para indexar las variables del sistema, pero puede ser usado durante el tiempo en que estén desconectadas las interrupciones y ninguna de las rutinas ROM se use antes de que vuelva a contener 23610 (5C3Ah).

Este libro tiene la intención de ser un manual de referencia, al que podrá acudir en busca de información mientras que utilice su Spectrum. He procurado no extenderme mucho en el texto principal, pero espero que encuentre en él la información que necesite. Obviamente, sería imposible detallar todas las rutinas de la ROM y todas sus utilidades, así que he seleccionado aquellas que considero de uso más frecuente y las más útiles. Más adelante, los programadores avanzados pueden leer el libro Complete Spectrum ROM Disassembly, del Dr. Ian Logan.


PUSH

2. Direcciones de llamadas útiles y cómo usarlas: la ROM de 16K

Es importante, antes de usar rutinas de la ROM, salvar siempre los registros H' y L' y devolverles su valor inicial antes de volver al BASIC. El registro IY debe contener siempre la dirección de la variable del sistema ERR NR 23610 (5C3Ah) cuando se usa una rutina de la ROM.

Impresión: RST 16 (10h)

El carácter cuyo código está en el registro A será impreso en la corriente que actualmente esté abierta. También se pueden usar los códigos de control de impresión (TAB, INK, OVER, etc.). Ver el manual del Spectrum para más detalles).

Abriendo y cerrando corrientes (para RST 16 (10h)): CALL 5633 (1601h)

CALL 5633 (1601h) hace que la dirección de salida usada por RST 16 (10h) sea el canal indicado por el contenido del registro A cuando se llama a esta rutina. Normalmente, A = 2 imprimirá en la pantalla principal, A = 3 lo hará por la impresora y A = 1 o imprimirá en la pantalla inferior. Con el Interfaz 1 conectado, otras corrientes pueden ser usadas para los Microdrives, red local u otros mecanismos. Los detalles de cómo desviar las corrientes para sus propios fines (por ejemplo: para controlar un interfaz con otros usos, tal como el Kempston u otro interfaz de impresora Centronics) los encontrará en la sección **Expandiendo palabras clave para salida**, en este mismo capítulo.

Detectando si la tecla Break está siendo pulsada: CALL 8020 (1F54h)

Esta llamada volverá con la bandera de acarreo puesta si la tecla BREAK no está siendo pulsada, y bajada si está siendo pulsada. *(Nota:* Esto comprueba si CAPS SHIFT y SPACE están siendo pulsadas simultáneamente.) Si usted quiere comprobar si sólo se pulsa SPACE, vea la sección **Tomando un carácter del teclado** para obtener más detalles; de momento puede usar:

```
A047    3E7F     10    LD     A, #7F
A049    DBFE     20    IN     A, (#OFE)
A04B    1F       30    RRA
A04C    D24CAO   40    JP     NC, PULSADA
```

Colocando la posición de impresión para usar con RST 16(10h): CALL 3545 (DD9h)

Esta rutina requiere que el registro B contenga el número de línea de la pantalla en la forma siguiente:

B = 24 - número de línea

(por ejemplo si B = 24, esta es la línea superior de la pantalla; si B = 1 es la línea inferior). Desafortunadamente, debido a una omisión en la escritura de la ROM, usted no puede usar las líneas 23 y 24 para imprimir en la pantalla principal; de modo que para usar la sección inferior debe establecer la corriente de salida para que RST 16 (10h) sea 1 y usar las dos líneas superiores de la pantalla inferior.

C = 33 - número de columna

(por ejemplo: si C = 33, esta es la columna más a la izquierda; si C = 2, es la más a la derecha).

Esta llamada actualiza automáticamente las variables del sistema con las posiciones de impresión en cualquier corriente que esté usando, como indicó al llamar a 5633 (1601h). Hay que tener cuidado al trabajar en la última línea de la pantalla principal, pues un mensaje scroll será generado después de imprimir en la última posición disponible. Esto producirá un retorno al BASIC, si se pulsa "n" o SPACE. Además, todo intento para imprimir en la corriente 1 causará un aumento de la pantalla inferior cuando el espacio disponible indicado por la variable del sistema DF SZ (23659) haya sido llenado. Esto puede producir algunos resultados inesperados.

Obviamente, cuando se usa la impresora, usted no puede indicar un número de línea; por tanto, no se usa el registro B.

Borrando la pantalla (toda la pantalla): CALL 3435 (D6Bh)

Esta llamada borrará toda la pantalla y reinicializará los atributos con los datos contenidos en la variable del sistema ATTR p para la pantalla principal y BORDCR para la pantalla inferior (23693 y 23624, respectivamente; ver el manual del Spectrum para más información).

Borrando la pantalla (pantalla inferior): CALL 3438 (D6Eh)

Esta rutina borrará solamente la pantalla inferior y reinicializa los atributos.

Nota: 3435 (D6Bh) y 3438 (D6Eh) reinicializan DF SZ a 2 y pueden corromper los datos del canal usado actualmente por RST 16 (10h), que, por tanto, necesitará ser reinicializado. Las posiciones actuales de impresión se colocan en la parte superior izquierda de las respectivas pantallas.

Desplazamiento vertical de la pantalla: CALL 3582 (DFEh)

Esta rutina desplazará la pantalla hacia arriba una línea, pero no afectará a la posición de la impresión actual. Por tanto, si imprime continuamente en la misma posición, desplazando el texto al final de cada línea, conseguirá un efecto parecido al de teclear en una máquina de escribir (la impresión tiene lugar en la parte inferior y el texto se desplaza hacia arriba cuando se hace un retorno de carro).

Dibujando en la pantalla: CALL 8933 (22E5h)

El punto indicado por el registro B (Y 0 - 175) y por el registro C (X 0 - A255) será dibujado en la pantalla. los colores de la tinta y el papel (INK y PAPER) pueden ser colocados previamente cargando las variables del sistema correspondiente con los atributos de los colores requeridos, o si no se usarán los actuales. OVER 1 puede ser colocado con SET 0, (IY + 87) o quitado con RES 0, (IY = 87) y cualquier punto se puede borrar con SET 2, (IY + 87).

Nota: Es perfectamente posible dibujar en la pantalla inferior, y esto puede ser anulado, si no se necesita, poniendo la corriente requerida con el CALL 5633 (1601h).

Sacando un número por una corriente

Esta es la rutina más complicada hasta ahora, y no hay una dirección de llamada única. Hay una rutina en la ROM que toma un número de 16 bits de dos direcciones y lo saca en forma decimal, pero esta rutina tiene la desventaja de que los bytes deben ser almacenados al contrario que como lo hace el Z80.

Esto significa que debe cargar el número en un par de registros, y a continuación volverlo a almacenar en memoria con el orden invertido (el byte más alto primero). Un programa corto como el que sigue valdría:

```
A06C    ED5B6CA0    1     LD     DE, (direccion_num)
A070    2170A0      9     LD     HL, MEMORIA_LIBRE
A073    72          17    LD     (HL), D
A074    23          25    INC    HL
A075    73          33    LD     (HL), E
```

La rutina de la ROM en 6696 (lA28h) puede ser llamada ahora con HL señalando a la dirección PALABRA TEMPORAL, que será sacada en ASCII por la corriente actual, pero hay dos inconvenientes:

1. Sólo saldrá correctamente si es menos que 10.000 decimal, y debe ser un entero.
2. El número saldrá con espacios por delante, de modo qué si se intenta unir todas las salidas para crear un número mayor, sólo funcionará si cada número de salida está entre 1000 y 9999; de otro modo serán introducidos espacios en medio.

Aquellos que tengan el Spectrum Pocket Book y hayan leído algo sobre esta dirección de llamada, pero no la hayan usado, estarán llamándome idiota. En este libro, el autor indica que el registro E controla cómo se formatea el número; pero si usted ha usado la rutina, sabrá que esto no es cierto, ya que el registro E es ignorado. Hay dos soluciones a este problema.

Primero podría escribir su propia rutina para saltarse la primera parte de la rutina de la ROM; de este modo:

```
A07B    D5        1     PUSH    DE
A07C    56        34    LD      D, (HL)
A07D    23        67    INC     HL
A07E    5E        100   LD      E, (HL)
A07F    E5        133   PUSH    HL
A080    EB        166   EX      DE, HL
A081    1E20      199   LD      E, #20
A083    C3301A    232   JP      #1A30
```

Nota: La última instrucción ha de ser un salto (JP), y la rutina que usted ha escrito ha de ser llamada con un CALL; de otro modo la pila se desorganizará, ya que la dirección de retorno está almacenada en ella y la rutina extrae (POP) lo que haya introducido (PUSH). Puede ver que el registro E es cargado en su rutina (borrando previamente cualquier dato que hubiera), y como ésta es un duplicado del principio de la rutina en ROM, usted puede ver ahora por qué el Spectrum Pocket Book está equivocado. Ahora puede cambiar la instrucción LD E, de esta rutina para que sea 48 decimal, en cuyo caso se imprimirán ceros delante del número, ó 255, que hará que no se imprima ninguna cifra no significativa.

"Pero, espere un minuto -estará diciendo usted-, ahora puedo ver por qué se debe cambiar el orden de los bytes." Correcto. Si se invierte el orden de carga del par de registros DE o, lo que es igual, se carga HL directamente, el número no ha de ser invertido; por tanto, vale la pena reescribir esta corta pieza de código para incorporarla, dentro de su propio programa, como una subrutina para imprimir números.

Hay otra parte de esta rutina en la ROM que merece ser considerada en este punto, y es la parte que comienza en 1A1Bh. Esta simplemente toma el número del par de registros BC y lo imprime sin ceros ni espacios, mucho más útil en algunos casos.

Con un pequeño trabajo adicional, estas rutinas le dan ahora la posibilidad de imprimir cualquier tipo de número desde su programa. Sin embargo, es probablemente más fácil usar las rutinas que toman un valor de la pila del calculador (para más detalles, ver el capítulo 8).

Tomando un carácter del teclado

La instrucción RST 56 (38h) es usada por el Spectrum al examinar el teclado y actualizar las variables del sistema FRAMES, y es llamada por la rutina de la interrupción enmascarable. Si todo lo que se quiere es ver si se ha pulsado una tecla, puede examinar el bit 5 de FLAGS 23611 (5C3Bh); si el bit está a 1, una tecla ha sido pulsada desde que fue puesto a 0 la última vez.

Téngase en cuenta que el bit 5 necesita ser puesto a 0 expresamente.

El código de la última tecla pulsada puede encontrarse en 23560 (5C08h) LAST K, detallado en el capítulo 4. Esto es suficiente para conseguir entrada desde el teclado, pero tiene desventajas.

Primero, es actualizado sólo 50 veces por segundo (60, si la frecuencia de la red es de 60 Hz, como en USA), de modo que usted no puede hacer una instrucción RES 5 y examinar inmediatamente el bit, ya que es casi seguro que el teclado no habrá sido examinado porque no se habrá producido una interrupción; por otra parte, si las interrupciones están inhabilitadas, el teclado nunca será examinado, a no ser que usted lo haga. Esto se puede realizar así:

```
A177    213B5C   10                  LD      HL, 23611  ;variable del sistema FLAGS
A17A    CBAE     20                  RES     5, (HL)
A17C    3EFF     30                  LD      A, #FF
A17E    32005C   40                  LD      (23552), A  ;23552 es parte de la variable del sistema KSTATE
A181    E5       50                  PUSH    HL
A182    FF       60                  RST     56
A183    El       70                  POP     HL
A184    AF       80                  XOR     A
A185    CB6E     90                  BIT     5, (HL)
A187    CA8DA1   100                 JP      Z, NINGUNA_TECLA
A18A     3A085C  110                 LD      A, (23560)   ;variable del sistema LAST KEY
A18D     00      120         NINGUN  NOP     ; lo que se quiera poner
```

Esta rutina devolverá en el registro A el código de la tecla que está siendo pulsada, o 0 si no hay ninguna. En la rutina anterior, la instrucción LD HL,23611 podría ser omitida, y en su lugar se podría examinar (IY + 1), ya que el registro IY contiene la dirección de la variable del sistema ERR NR (23610) (5C3Ah) y es usado por la ROM para direccionar las variables del sistema.

Debido a esto, si usa el registro IY en sus programas, debe asegurarse de que las interrupciones están inhabilitadas o dirigidas a sus propias rutinas, y debe reinicializar también el registro IY a la dirección correcta antes de usar la ROM o de colocar las interrupciones en su estado normal.

El segundo problema es que usted no puede examinar más de una tecla pulsada a la vez. Esto es debido a qué si una combinación de teclas es pulsada, y no es ningún carácter válido, el Spectrum la ignorará.

Para evitar esto, necesitará escribir su propia rutina para examinar el teclado. En primer lugar, si usted sólo quiere saber si una tecla está siendo pulsada, sin preocuparse de cuál o cuántas teclas son, podría hacer algo así:

```
A16B    AF       10         XOR      A          ; se asegura que A contiene 0 para explorar todas las teclas
A16C    DBFE     20         IN       A, (#FE)   ; FE es el puerto de entrada del teclado y otros.
A16E    161F     30         LD       D, 31
A170    A2       40         AND      D          ;elimina los otros (3 bits altos)
A171    AA       50         XOR      D          ;invierte los bits restantes
A172    28FE     60         JR       Z, NINGUNA_TECLA
```

El teclado puede ser examinado más detalladamente, para descubrir qué teclas están siendo pulsadas, cargando el registro A con el valor de las líneas a examinar antes de la instrucción IN A, (FEH).

El capítulo 23 del manual del Spectrum explica cómo está dispuesto el teclado, y también se señala en el apéndice D de este libro, pero el esquema resumido es el siguiente. El primer carácter, en todos los casos, es el BIT 0, y el último el BIT 4; el número hexadecimal ha de ponerse en el registro A para poner a 1 el bit de la línea que se quiere examinar:

CAPS SHIFT-V	FEh
A-G	FDh
Q-T	FBh
1-5	F7h
0-6	EFh
P-Y	DFh
ENTER-H	BFh
SPACE-B	7Fh

Así, por ejemplo, para comprobar si se está pulsando la tecla ENTER, se usaría el siguiente programa:

```
A050    3EBF    1    LD     A, #OBF
 A052   DBFE    2    IN     A, (#0FE)
A054    E601    3    AND    1
 A056   28FE    4    JR     Z, ENTER_PULSADA
```

Para ver si se está pulsando más de una tecla, puede usar los operadores lógicos AND, OR, etc., si están en la misma línea, o las instrucciones, para comprobar bits. Vea a continuación CALL 5598 (15DEh) para conseguir fácilmente caracteres normales.

Esperar entrada: CALL 5598 (15DEh)

Esta es una de las rutinas más útiles, permite la entrada desde cualquier corriente con dirección de entrada. Antes de usar la corriente desde la cual se pide entrada, debe abrirse con la rutina en 5633, descrita anteriormente. Cuando la rutina Esperar entrada es llamada (CALL), llama a continuación a la rutina de entrada del canal actual. Cuando se vuelve de ésta se comprueba la bandera de acarreo; si está puesta, el control se devuelve al programa principal. Si la bandera de acarreo no está puesta, se comprueba la bandera 0, y si está puesta, el proceso se repite.

La rutina es usada por el Spectrum como control de la subrutina de entrada actual, normalmente la entrada por teclado; pero si CURCHL está puesto señalando a un canal con la dirección de entrada apuntando a su propia subrutina, esta será usada (cómo hacer esto se explica en el capítulo 5). Cuando se utiliza con la corriente 1 (el teclado y la pantalla inferior), la rutina esperará a que una tecla sea pulsada, y devuelve su código en el registro A. Este uso se demuestra en el programa DeBASE del apéndice G.

Hay, sin embargo, un problema para leer el teclado cuando el modo está cambiado. Cada vez que la rutina es llamada, TV FLAG 23612 (5C3Ch) IY + 2 es comprobada, y si el bit 3 está puesto, la memoria intermedia (buffer) de la entrada se copia en el área de edición de la pantalla. Esto se soluciona usando directamente la rutina Entrada de Tecla en 4264 (10A8h) y no por medio de la rutina Esperar Entrada, como se muestra en el programa DeBASE en el apéndice G en la etiqueta INPUT. El programa duplica la rutina Esperar Entrada, pero comprueba que la bandera se usa en la etiqueta INPUTF, y si ha introducido el programa (o esta sección del mismo) el problema puede demostrarse pulsando CAPS SHIFT y SYMBOL SHIFT para cambiar al MODO EXTENDIDO. La última línea que fue tecleada aparecerá en la parte inferior de la pantalla.

Cuando se usan estas rutinas para leer el teclado, las interrupciones deben estar permitidas, y la rutina normalmente utilizada para las interrupciones en 56 (38h) debe ser llamada dentro del ciclo de interrupción, si no, no se recibirá ningún dato de entrada.

Copia de la pantalla a impresora: CALL 3756 (EACh)

Esta rutina no requiere inicialización, y con un simple CALL saldrá una copia de la pantalla en la impresora ZX.

Impresión de gráficos por impresora: CALL 3789 (ECDh)

Esta rutina es similar a la de copia de pantalla, en que usa el contenido de la memoria intermedia de la impresora y lo imprime. Es usada por la rutina RST 16, la cual trata, normalmente, a la memoria intermedia como una línea de pantalla (ocho puntos de altura). Si usted coloca sus gráficos en la memoria intermedia una línea cada vez y después hace una llamada (CALL) a esta dirección, la rutina copiará la memoria intermedia por la impresora.

Nota: El conjunto consta de 32 bytes por línea de puntos; la siguiente debe ir a continuación, y no como en la pantalla. La memoria intermedia es puesta a 0 después de ejecutar esta rutina.

Borrando la memoria intermedia de la impresora: CALL 3807 (EDFh)

Simplemente, llena la memoria intermedia con ceros.

Usando el Beep: CALL 949 (3B5h)

La pareja de registros DE contiene la duración, y HL la frecuencia; 0 es una frecuencia alta, y FFFFh es baja. El problema de esta rutina es que la duración depende de la frecuencia; por consiguiente, si divide por la mitad la frecuencia, se dobla la duración para un valor dado en DE. Los valores necesarios se calculan como sigue:

HL = (437 500/frecuencia) - 30,125
DE = duración * frecuencia (la duración deberá ser en segundos)

La razón de que el 30,125 esté restado del cálculo de HL es que la rutina utiliza 120,5 T estados para generar la nota y modificar sus propios registros, etc.

El Do medio es, aproximadamente, 261 hertzios; por tanto, el valor de HL sería, aproximadamente, 1.646 en decimal, y para que la duración fuera un segundo, DE sería 261 en decimal.

Recuerde que las interrupciones de la ULA se producirán 50 veces por segundo (60 veces en Estados Unidos), y éstas distorsionarán la salida del sonido si la rutina está en los 16K inferiores de la RAM.

Imprimiendo mensajes: CALL 3082 (COAh)

En la entrada, la pareja de registros DE debe contener la dirección de la marca de inicio de la tabla de mensajes; dicha marca debe tener el BIT 7 alto y el registro A debe contener el número de entrada en la tabla de mensajes, cada uno de los cuales debe tener el BIT 7 a 1 en el último byte. El primer mensaje tiene el número 0.

Si quiere imprimir el mensaje "yo soy" podría tener en su programa una línea tal como: MENSAJE DEFM "yo soy"; pero esto haría que se imprimiese todo lo que se encontrase a continuación, hasta que apareciese un byte con el bit 7 alto. Usted debe cambiar, por tanto, la línea por 'MENSAJE DEFB 80H: DEFM "yo soy": DEFB"" + 80H'. La rutina restará el 80H e

imprimirá el último espacio correctamente, pero sabrá que ha llegado al final del texto y devolverá el control a la rutina que la ha llamado. Por tanto, el programa completo puede ser algo así:

```
A14E    3E00      10             LD     A, 0    ;para imprimir el primer mensaje de la tabla.
A150    CD53A1    20             CALL   PR_MES
                  30     ; . . .
                  40     ; EL RESTO DEL PROGRAMA VA AQUI
                  50     ; . . .
A153    115AA1    60     PR_MES  LD     DE, MENSAJ
A156    CD0A0C    70             CALL   3082
A159    C9        80             RET
A15A    80        90     MENSAJ  DEFB   #80
A15B    454E5452  100            DEFM   "ENTRADA"
A162    A0        110            DEFB   " "+#80
A163    454E5452  120            DEFM   "ENTRADA 2"
A16C    A0        130            DEFB   " "+#80
```

Esto imprimirá en la posición actual de la corriente en uso el mensaje <ENTRADA>. Para imprimir <ENTRADA 2>, el registro A debe contener 1. Los símbolos mayor-menor marcan los límites de lo que se imprime, y no aparecerán cuando se ejecute el programa. Esta es la rutina usada por la ROM para expandir las palabras-clave y generar mensajes de error cuando desde otra rutina se llama a 2898 (B52h). El programa DeBASE (ver apéndice G) hace amplio uso de esta rutina.

Expandiendo claves para la salida: CALL 2898 (B52h)

Siempre que el Spectrum encuentra un código especial (los que tienen el bit 7 alto) tiene que decidir qué hace con él, ya que puede ser un gráfico definido por el usuario, un gráfico de bloques o una palabra del vocabulario BASIC.
Normalmente, esto se realiza automáticamente cuando se usa RST 16 (10h) (la cual, a su vez, usa esta rutina). Si ha cambiado la dirección de la salida en una corriente (por ejemplo, para enviar al interfaz de su propia impresora), cuando use su rutina y el registro A contenga el código de una clave y desee expandirlo tendrá que hacerlo usted mismo. Si lo deja como está, conseguirá algunos resultados sorprendentes en la impresora. Esta rutina puede usarse para expandir los comandos del BASIC, pero tendrá que manejar los gráficos por sí mismo y asegurarse de que los códigos correspondientes no sean enviados a la rutina; si no, la impresora quedaría bloqueada.

Nota: Cuando está expandiendo, la rutina hace repetidas llamadas a la dirección apuntada por la corriente en uso y retorna sólo después de que todas las letras hayan sido sacadas. Esto significa que debe hacer un salto a la rutina (JP) para asegurarse de que el código en el registro A no es impreso cuando se devuelve el control al final de la impresión de una palabra BASIC. Esto se demuestra en el siguiente programa típico que envía la salida a una impresora:

```
A545    2A4F5C    10     INICIO  LD     HL, (23631); CHANS (dirección del canal de datos)
A548    010FOO    20             LD     BC, 15; incremento para el canal 3 (impresora)
A54B    09        30             ADD    HL, BC; HL indica ahora la posición que contiene la dirección
                  40                            ; a ser llamada cuando se realiza la salida por este canal
A54C    0153A5    50             LD     BC, EMPIEC; EMPIEC=inicio de su rutina de impresora
A54F    71        60             LD     (HL), C
A550    23        70             INC    HL
A551    70        80             LD     (HL), B
A552    C9        90             RET    ; canal 3 inicializado hacia su rutina.
A553    47        100    EMPIEC LD      B, A; salva el código ASCII en B
A554    FEA5      110            CP     165
A556    D2520B    120            JP     NC, #B52; rutina de expansión en ROM
A559    FE0D      130            CP     13
A55B    28FE      140            JR     Z, CRLF; salto de línea (Carriage Return y Line Feed)
A55D    FE20      150            CP      32
A55F    D8        160            RET    C; cualquier código inferior a 32 no es imprimible
A560    FE80      170            CP     128
A562    3800      180            JR     C, IMPRIM; es un carácter normal
```

```
        190                            ; cualquier cosa que llegue
        195                            ; aquí es un gráfico y lo debe tratar como usted quiera.
A564    00      210     IMPRIM  NOP    ; su rutina de impresión empieza aquí.
A565    C9      220             RET    ; vuelve a por el siguiente carácter, si no hay ninguno
                230                    ; el control vuelve al programa original.
```

Este programa es uno de los más elementales; si desea enviar cualquier código de control de la impresora, la rutina no lo enviará. Para evitar esto, deberá realizar más comprobaciones y actuar en consecuencia. La primera sección sólo necesita ser llamada una vez, pues modifica la dirección de la salida a la corriente 3. Sólo algunas veces necesita reinicialización después de un comando NEW desde el teclado o si la dirección de salida ha sido cambiada por otra parte del programa.

Expansión de gráficos de bloque: CALL 2878 (B3Eh)

Si tiene el código de un gráfico de bloques y quiere convertirlo en el gráfico en sí, la rutina en 2878 (B3Eh) lo hará por usted. Al llamarla, la dirección de base de los ocho bytes disponibles, donde quiere construir el gráfico, debe estar en la pareja de registros HL, y el registro B debe contener el código del gráfico de bloque. Se necesitan dos llamadas (CALL), la segunda inmediatamente después de la primera, ya que cada CALL construye cuatro bytes del carácter.

El primer byte de su bloque de ocho será la parte superior del gráfico, y HL apuntará al byte siguiente después del último del gráfico. Las parejas de registros alteradas por esta rutina son AF, HL y BC; no usa ninguna otra.

Dibujando círculos: CALL 9005 (232Dh)

La rutina, para dibujar un círculo, necesita que los parámetros de éste estén en la pila del calculador del Spectrum. De este modo, lo primero que hay que hacer es ponerlos en la pila del calculador. Una rutina localizada en 11560 (2D28h) lo hará por nosotros si ponemos el número que deseamos apilar en el registro A (ver en capítulo 8 Uso del calculador). Todo lo que necesita saber ahora es que esta rutina reinicializa el registro IY para apuntar a ERR NR y que destruye el contenido de la mayoría de los otros registros, de forma que usted debe salvarlos antes de llamar esta rutina.

Los parámetros deben ser salvados en la pila del calculador en el orden X, Y, Z, (Z = radio). La rutina para dibujar el círculo actualiza las variables del sistema COORDS, de modo que si usted no quiere que cambien debe salvarlas antes de dibujar el círculo y restituirlas después. De este modo, la rutina para dibujar un círculo quedaría así:

```
A1D3    2A7D5C   10     LD      HL, (23677); COORDS (coordenadas)
A1D6    E5       20     PUSH    HL;salva coordenadas
A1D7    3AD7A1   30     LD      A, (X); donde X esta entre 0 y 255
A1DA    CD282D   40     CALL    #2D28; rutina en la ROM que apila
A1DD    3ADDA1   50     LD      A, (Y)
A1E0    CD282D   60     CALL    #2D28
A1E3    3AE3A1   70     LD      A, (Z); Z=radio (debe caber en la pantalla o tendrá error)
A1E6    CD282D   80     CALL    #2D28
A1E9    CD2D23   90     CALL    #232D; dibujar circulo
A1EC    E1       100    POP     HL
A1ED    227D5C   110    LD      (23677), HL; devuelve las coordenadas originales
```

Si usted quiere dibujar su círculo en la posición actual indicada por COORDS, deberá colocarla en la pila del calculador, pero tendrá que volver a guardarla en la pila de la máquina si no quiere que se modifiquen.

Dibujando una línea: CALL 9146 (23BAh)

La rutina para dibujar una línea empieza propiamente en 9399 (24B7h) y toma sus parámetros de la pila del calculador del mismo modo que la rutina del círculo. En este caso, es fácil evitar la parte que usa la pila del calculador.

La rutina toma el punto de inicio en COORDS, así que si usted quiere partir desde alguna otra posición tendrá que cargar COORDS con su posición de inicio después de salvar su contenido, por si lo necesita posteriormente. De otro modo, COORDS contendrá la dirección del final de su línea. Al llamarla, el par de registros DE debe contener los signos de los parámetros del comando DRAW, que, a su vez, deben estar guardados en los registros BC. Deben ser - 1 (FFh) para negativo y + 1 (01h) para positivo. En los registros C y E se guarda X, mientras que en B y D se guarda Y.

La rutina en lenguaje máquina equivalente a la línea DRAW 0,175 del BASIC, sería así:

```
A0A7    0100AF      LD      BC, #AF00; 175,0
A0AA    110101      LD      DE, #0101; +
A0AD    CDBA24      CALL #24BA; lo que es equivalente a DRAW -255,0
A0B0    01FF00      LD      BC, #00FF; 0,255
A0B3    11FF01      LD      DE, #01FF; +, -
```

Fíjese que esta rutina dibujará desde la posición actual COORDS y que no se ha intentado modificarla o guardarla. Después de ejecutar la rutina, estos parámetros señalarán al último punto trazado mientras se dibujaba la línea. y se producirá un error si intenta dibujar fuera de la pantalla.

Encontrar la dirección de un punto de la pantalla: CALL 8874 (22AAh)

Para encontrar la dirección del byte que contiene el punto de la pantalla usado por un comando PLOT se puede llamar a esta rutina con la pareja de registros BC conteniendo las coordenadas X e Y (Y 0 - 175 en B, y X 0 - 255 en C) y volverá con la pareja de registros HL conteniendo la dirección y el registro A conteniendo la posición del bit.

Borrando parte de la pantalla: CALL 3652 (E44h)

Esta rutina borrará el número de líneas indicado por el registro B desde la parte inferior de la pantalla (por ejemplo: si B contiene 1, entonces sólo se borra la línea inferior, y si B contiene 10, entonces se borran las diez líneas inferiores). La parte inferior de la pantalla es siempre la línea 24. no la línea más baja de la pantalla principal.

Desplazando hacia arriba parte de la pantalla: CALL 3584 (E00h)

Para desplazar hacia arriba parte de la pantalla se debe colocar primero en el registro B el número de líneas a desplazar menos uno. Después de esto se puede llamar a la rutina. Después de cada llamada, la línea inferior será borrada y se deben mover, al menos, dos líneas. De nuevo, las líneas se cuentan desde la parte inferior de la pantalla completa.

Entrada desde el canal actual: CALL 5606 (15E6h)

Esta rutina coge la dirección del canal actual de la variable del sistema CURCHL, encuentra la dirección de la subrutina de entrada en el área de información del canal y entonces la llama.

Borrando la pila del calculador y el área de trabajo: CALL 5823 (16BFh)

Esta rutina puede ser muy útil para asegurarse de que el calculador tiene la máxima área de trabajo y que no hay ningún dato inútil en la pila del calculador. Sólo usa la pareja de registros HL, y volverá con el contenido de la variable del sistema STKEND en ellos.

Salvando, cargando y verificando

Las rutinas SAVE y LOAD del Spectrum son muy concretas y fáciles de usar si ambas operaciones, salvar y cargar, se van a realizar desde dentro de un programa, si la longitud exacta de los datos es conocida y si no importa si el control de la máquina se devuelve al BASIC en el caso de un error o de que se pulse BREAK. Si la longitud no se conoce o lo que se maneja es algo que no son datos, la cosa se vuelve más complicada.

Normalmente, cuando está cargando, el Spectrum espera recibir una cabecera antes del bloque principal, y es esta cabecera la que dice al Spectrum cómo manejar el bloque principal de datos que viene a continuación. Es más fácil salvar y cargar sin la cabecera, pero, como señalamos anteriormente, esto sólo puede hacerse si los parámetros exactos son conocidos.

La cabecera es de 19 bytes de largo (no 17, como muchos libros dicen), pero sólo los 17 bytes de en medio necesitan ser actualizados, ya que las rutinas que salvan y cargan crean el primero y el último ellas mismas.

El byte 1 es siempre 00 para una cabecera, y el último byte es un byte de paridad, que es creado por la rutina, así que no hay necesidad de preocuparse de él. El byte 2 indica el tipo de bloque principal:

0 para un programa BASIC
1 para una matriz numérica
2 para una matriz alfanumérica
3 para un bloque de bytes

Los bytes del 3 al 12 almacenan el nombre. Los bytes 13 y 14 almacenan la longitud del bloque principal. En un programa BASIC ésta sería la diferencia entre las variables del sistema E LINE-PROG. Los bytes 15 y 16, en un bloque de bytes, almacenarán la dirección de inicio donde se carga el bloque. Estos, en un programa BASIC, almacenarán el número de línea de inicio automático si existe o la dirección de inicio si no existe. Para una matriz, el byte 16 almacena el nombre de la matriz en la forma de:

Bits del 0 al 4, el nombre (A = 1 a Z = 26).
Bit 5 a 0, si la matriz es numérica.
Bit 6 a 1, si la matriz es alfanumérica.
Bit 7, siempre a 1.

Los bytes 17 y 18 almacenan la longitud del programa, si lo que estamos tratando es un programa BASIC (es decir, la diferencia entre las variables del sistema VARS - PROG). El último byte es de paridad, y se crea al llamar a la rutina.

La cabecera

BYTES 1 2 3....... 12 13 14 15 16 17 18 19

BANDERA	TIPO	NOMBRE PROGRAMA	LONGITUD DATOS	INICIO	LONGITUD PROGRAMA	PARIDAD

IX+ 00 1........ 10 11 12 13 14 15 16 17

Para salvar en cinta una cabecera, ésta debe ser creada como se ha dicho y salvada seguida por el bloque de datos principal, o si los parámetros del bloque de datos principal se conocen y siguen siendo conocidos cuando se vaya a cargar, sólo hace falta que se salve el bloque de datos.

Hay varios puntos de entrada a la rutina SAVE que se pueden usar, teniendo cada uno sus pros y sus contras. El primero a tener en cuenta es quizás el más fácil, pero también el más problemático. Para usarlo se debe almacenar en el registro IX el punto de inicio de la cabecera (con el byte 2, como se ha descrito en la información de cabecera anterior) y la pareja de registros HL debe señalar a la dirección de inicio del bloque principal que se va a salvar. Una vez que estos registros han sido modificados, se debe realizar una llamada (CALL) a la dirección 2416 (0970h), y entonces se salvarán ambos, la cabecera y el bloque principal. Los problemas son los siguientes:

1. El mensaje Start tape then press any key aparecerá. Si se pulsa cualquier tecla, excepto BREAK, todo irá bien; pero si no, el control será devuelto al BASIC mediante la rutina de manejo de error.
2. La tecla BREAK se comprueba periódicamente durante la ejecución de esta rutina SAVE. Si se pulsa, provoca un retorno prematuro a la rutina de manejo de error del BASIC, lo cual puede ser molesto.
3. La cabecera se salva de forma que asegure que la información salvada pueda ser vuelta a cargar desde el BASIC. Según las circunstancias, esto puede considerarse como una ventaja, no como un problema.

El siguiente punto de entrada puede considerarse similar al anterior, con la excepción de que no pregunta ni espera a que se pulse una tecla. A éste se accede mediante una subrutina en el programa principal, porque el funcionamiento correcto depende de que se hayan introducido los valores adecuados en la pila de la máquina. Primero se inicializan las parejas de registro IX y HL, como hemos indicado antes, entonces se llama (CALL) a la siguiente rutina desde el programa principal :

```
A035  E5         1 SAVE    PUSH  HL
A036  C38409     2         JP    2436  ; (#0984)
```

Después de que se haya realizado la operación de salvar el control se devolverá a la dirección siguiente a la del punto en que se llamó a esta rutina. Este es un método sencillo de salvar bloques sucesivos completos con cabeceras, evitando repetir los mensajes start tape.

Para realizar la operación sin que sea posible pararla y que se devuelva el control al BASIC, el comienzo normal debe ser evitado y la cabecera y el bloque de datos deben ser salvados como bloques separados. La rutina SA_BYTES, situada en 1218 (04C2h), se llama, normalmente, al hacer algún SAVE con el registro A conteniendo 00, para una cabecera, o FFh, para un bloque de datos. Lo primero que esta rutina hace es cargar la píla de máquina con la dirección de la rutina SAVE/LOAD RETURN. Esta rutina permite las interrupciones y mira si la tecla BREAK ha sido pulsada. La rutina de manejo de error es llamada por medio de un RST 8, si BREAK ha sido pulsada, volviendo el control, por tanto, al BASIC. Si BREAK no ha sido pulsada, el control se devolverá a la dirección original de RETORNO, introducida dentro de la pila por la rutina que ha realizado la llamada.

Si no se ha colocado la dirección de retorno de la rutina SAVE/LOAD, la dirección de retorno que estará en la pila de la máquina es la del programa que ha realizado la llamada, y el control se le devolverá a la salida de la rutina SAVE, con la bandera de acarreo bajada, si se hubiera intentado parar el programa por medio de la tecla BREAK, y puesta a 1, en caso contrario, y las interrupciones estarán desconectadas: por tanto, éstas deben ser permitidas de nuevo. También deberá escribir algún tipo de mensaje para asegurarse de que la cinta ha sido puesta en marcha;

por ejemplo, usando las rutinas de impresión de mensaje y Esperar Tecla detalladas en este mismo capítulo.

Para salvar un bloque de código por este método, el registro IX debe señalar al inicio del bloque, y el registro DE debe contener la longitud. El registro A debe contener FFh, para un bloque de datos, ó 00, para una cabecera. A continuación, se debe realizar una llamada directa a la dirección 1222 (04C6h). Recuerde que la pareja de registros DE ha de contener 17 (11h) si se está salvando una cabecera estándar. Con este método se pueden salvar bloques de datos sin cabecera, pero sólo se pueden cargar posteriormente si se conoce la longitud.

Si un retorno al BASIC no nos preocupa, o es una ventaja, entonces se puede usar la misma rutina, pero desde el inicio, que está en 1218 (04C2h). Si se pulsa BREAK, el control volverá al BASIC. De nuevo tendrá que asegurarse de que la cinta está funcionando.

Cargando y verificando

Se pueden pasar datos de la cinta al Spectrum de dos formas: con cabecera o sin ella. Cuando hay cabecera, ésta puede ser usada para proporcionar todos los parámetros para la carga del bloque de datos principal que sigue o, como cuando se carga desde el BASIC, para averiguar los detalles que son desconocidos, para comprobar que los conocidos son correctos, o para asegurarse de que se está cargando el bloque de datos correcto. Si no hay cabecera, todos los detalles que habían estado en esta sección deben ser conocidos antes de que el bloque de datos pueda ser cargado.

Es posible crear un tipo diferente de cabecera del usado por el Spectrum para definir los parámetros del bloque que viene a continuación, salvando un bloque de datos de longitud fija y escribiendo su propia rutina de decodificación, la cual proporcionará los detalles para cargar el bloque principal. Esta es una buena idea que previene la posibilidad de perder un bloque, porque usted olvidó la longitud o dónde estaba en la cinta y, por tanto, no puede cargarlo.

Esto también previene que el código sea cargado por un usuario "no autorizado", a no ser que él se tome la molestia de escribir su propio programa para examinar qué ha hecho usted y logre interpretarlo. Una rutina que crea una cabecera especial se da en el programa DeBASE en el apéndice G.

Para cargar un bloque de datos precedido por una cabecera normal, lo primero que se debe hacer es asignar un espacio de 34 bytes en la memoria. En los primeros 17 bytes se debe crear una cabecera "ficticia ". Esta define detalles que es necesario que coincidan antes de que el bloque de datos sea cargado.

Una vez que estos detalles coinciden, cualquier información no coincidente será comparada con la cabecera real y, suponiendo que los cambios sean aceptables, estos detalles necesarios para la carga serán tomados de la cabecera "ficticia". Si las diferencias no son aceptables, se genera un error del BASIC por medio de un RST 8. Los segundos 17 bytes serán rellenados con la cabecera de la cinta, para comparar; sólo cuando coinciden, este área volverá a estar libre para su uso.

La cabecera ficticia se construye de la misma manera que la cabecera salvada en la cinta, y de nuevo, el primero y el último byte del total de 19 se generan internamente, de modo que no están incluidos en los 17 bytes de la especificación. El primer byte debe ser igual al primer byte de la cabecera de datos que se va a cargar (el tipo debe coincidir). Este está contenido en el registro A cuando se llama a la subrutina LOAD BYTES, y sólo necesita ser puesto si la llamada se realiza directamente. Si los tipos no casan, se espera un nuevo bloque para repetir el proceso hasta que los tipos coincidan. El segundo byte (el primero de los 17 con los que se ha

construido la cabecera ficticia) debe coincidir de nuevo: 0, para un programa en BASIC; 1, para una matriz numérica, etc. Si no coinciden, se esperará al próximo bloque. Los siguientes 10 bytes son el nombre. Si no importa el nombre, entonces el primero de estos bytes debe ser FFh, los nombres deben coincidir. Los dos bytes siguientes son la longitud; si éstos están puestos a 0, la longitud será tomada desde la cabecera de la cinta; de otro modo, las dos longitudes deben coincidir. Para un bloque de códigos, los bytes 15 y 16 serán la dirección de inicio para cargar, mientras que para una matriz en BASIC el byte 15 será ignorado y el byte 16 será el nombre de la matriz, de la misma forma que cuando se realizó el SAVE. Para un programa BASIC, el byte 15 será 0, y el byte 16 será 80h. El último byte puede ser ignorado.

Antes de intentar usar la cabecera ficticia, el byte bajo de la variable del sistema T ADDR debe contener 0,1, para realizar un LOAD, o un 0,2, para un VERIFY.

La pareja de registros HL debe contener la dirección a partir de la cual se va a almacenar el bloque principal ó 0 si se usa la información en la cabecera de la cinta. Para una matriz de BASIC, ésta será el inicio de la matriz de datos, a continuación, el nombre de la matriz y la longitud en el área de variables de un programa BASIC.

Finalmente, el registro IX ha de señalar a la dirección del primer byte de la cabecera ficticia. La rutina ROM, que empieza en 1889 (761h), puede ser llamada ahora para cargar o verificar tanto la cabecera como el bloque de datos principal que están en la cinta.

La carga o verificación sin una cabecera pueden ser conseguidas sólo si todos los parámetros del bloque que se va a leer se conocen. Primeramente, el registro A se carga con FFh, lo que indica que se va a cargar un bloque de datos. Después, la pareja de registros DE se cargan con la longitud total del bloque a leer. Entonces, el registro IX es colocado con la dirección en la que se van a almacenar los datos leídos o verificados. Esto debe ir acompañado por la bandera de acarreo puesta a 1, si se hace un LOAD, o a 0, para un VERIFY. Finalmente, se llama a la rutina de la ROM en 1366 (556h) para realizar la operación.

La parte inicial de esta rutina carga previamente de la pila de la máquina del mismo modo que la rutina SAVE ya comentada, volviendo con una indicación de error si se pulsa la tecla BREAK. Esto puede ser evitado llamando la siguiente subrutina que debe estar almacenada en su propio programa:

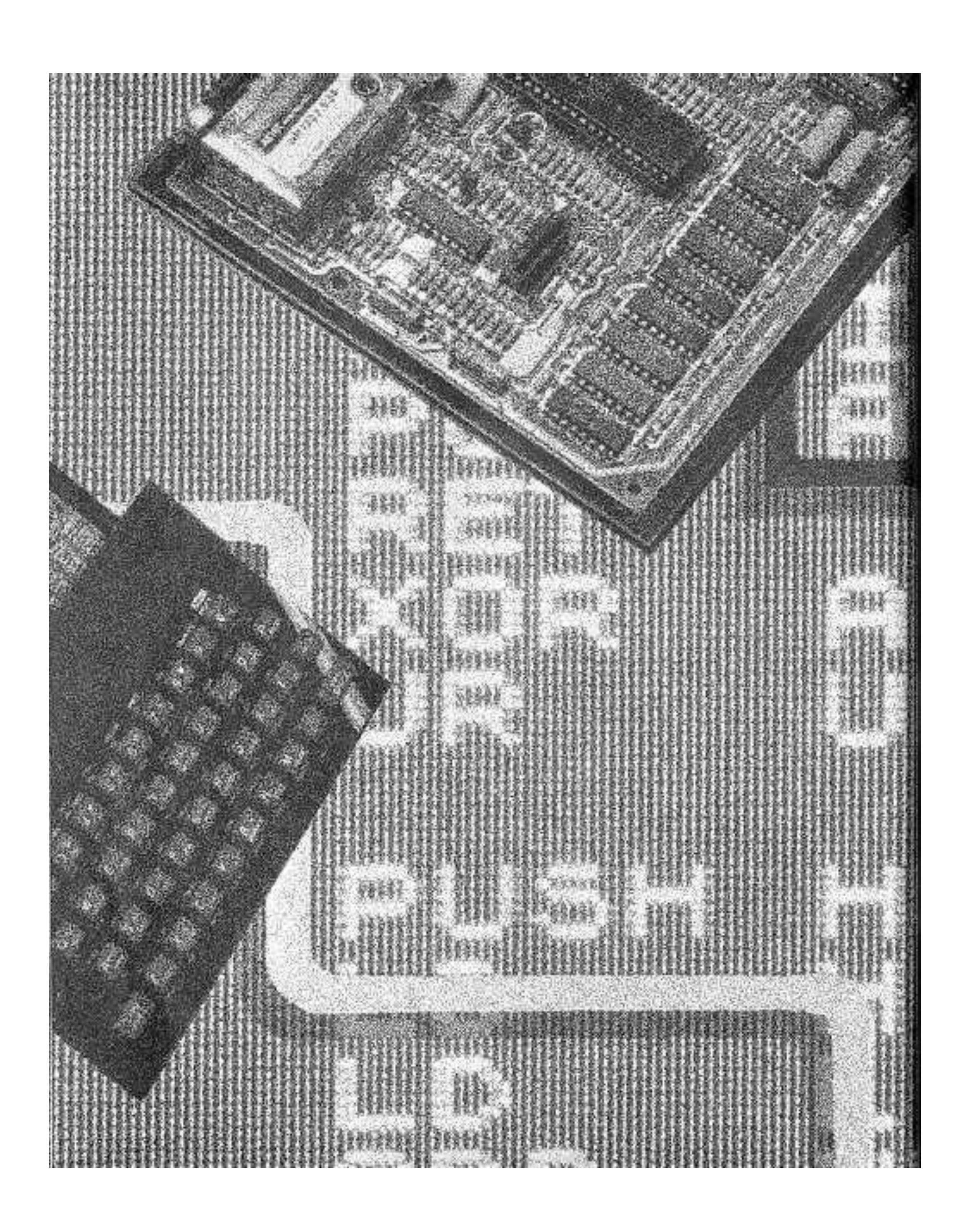
```
A1F7   14            CARGA  INC   D; baja la bandera cero
A1F8   08                   EX    AF, AF'
A1F9   15                   DEC   D; recupera el valor original de D
A1FA   F3                   DI    ; las interrupciones deben estar desconectadas para que los
                                  ; tiempos sean constantes, No son vueltas a conectar por la
                                  ; rutina en ROM al terminar,
A1FB   3E0F                 LD    A, #0F
A1FD   D3FE                 OUT   (#FE), A;  pone a 1 el borde y el conector EAR
A1FF   C36205               JP    #562; salta a la rutina principal de carga.
```

La instrucción OUT pone el color del borde blanco con los tres bits bajos; esto puede cambiarse cuando se requiera un color diferente. El bit 5 debe ser dejado a 1, ya que inicializa el puerto EAR para recibir las señales que vengan de la cinta.

Si se produce un error en la carga de la cinta, puede ser detectado por la bandera de acarreo puesta a 0 cuando se devuelve el control a la rutina que efectúa la llamada, ya que, si se ha pulsado BREAK, produciría un retorno inmediato; si se quiere comprobar esto se debe llamar a la rutina de comprobación de la tecla BREAK, después de comprobar si hay algún error.

Nota: Las interrupciones estarán desconectadas al terminar la carga.

La rutina puede ser usada también para leer una cabecera, en lugar de un bloque de datos, cargando el registro A con 0 antes de llamarla. Esto puede ser útil si desea crear un tipo diferente de cabecera y leerlo con su propia rutina, como se demuestra en el programa DeBASE en el apéndice G.


PUSH

3. La ROM del Interfaz de 8K

Con la adición de la ROM de 8K en el interfaz del microdrive, Sinclair ha presentado la posibilidad de expansión en la forma de una ROM suplementaria, como se usa en el microordenador BBC, así como una vía fácil de expansión del BASIC Sinclair. Lo primero que necesita comprender es el mecanismo por el cual la CPU Z80 puede direccionar la memoria situada fuera del rango normal de 64 kilo bytes.

La CPU tiene dos grupos de líneas de información -A 1-16 (que usa para decirle a la memoria qué byte se va a emplear)- llamada bus de direcciones y D 0- 7, bus de datos, que se usan para leer y escribir a la dirección de memoria indicada por el bus de direcciones. Normalmente, es imposible el acceso fuera del rango de 64K, de modo que se debe realizar algún tipo de cambio que sustituya un banco de memoria en el mismo rango de direcciones.

Esto se hace fácilmente eligiendo un byte de la memoria, mirando si aparece su dirección en el bus de direcciones y, cuando esto suceda, cambiando a la memoria alternativa. La ejecución continuará entonces en la dirección siguiente, ya que el contador del programa se incrementará para apuntar a la dirección siguiente, como es normal; pero el dato será cogido desde el nuevo banco de memoria.

En el Spectrum esto se hace comprobando en el bus de direcciones si aparece la dirección 8, que es la rutina de manejo de errores. Dado que esta dirección sólo es generada por un RST 8, el tope de la pila contendrá siempre la dirección de la instrucción siguiente a la de llamada (es decir, la dirección de retorno). La rutina de la ROM suplementaria toma esta dirección y examina su contenido. Los valores en el campo 00 al 1Ah causan un retorno a la ROM de 16K, pues éstos son códigos normales de error, pero los números entre 1Bh y 32h son usados como "código de enlace" (hook code). Estos llaman a las rutinas de la ROM suplementaria que se detallan ahora. Para usar un código de enlace, use RST 8 seguido por el código de enlace definido con DEFB. Todas las direcciones se refieren a la ROM suplementaria.

Entradas

Leer tecla: Código de enlace 1Bh, Dirección 6617 (19D9h)

Es similar al comando GET en algunos BASIC (aunque el BASIC de Sinclair no tiene este comando). Espera hasta que una tecla es pulsada, y entonces vuelve con el código de ésta en el registro A. La interrupción enmascarable debe estar activa, ya que la que usa la rutina de la ROM de 16 K se utiliza para explorar el teclado.

Entrada por el RS 232: Código de enlace 1Dh, Dirección 2945 (B81h)

Para usar esta rutina, primero se debe fijar la velocidad de transmisión usando la variable del sistema BAUD 23747/8 (5CC3/4h), calculada como 3.500.000/(26 * velocidad de transmisión) - 2, siendo 3.500.000 la frecuencia del reloj del Spectrum. Después SER_FL 23 751 (5CC7h) ha de ponerse a 0, y entonces se puede llamar a la rutina de entrada. El registro A contendrá el código del carácter recibido, y la bandera de acarreo estará puesta a 1. La rutina esperará cierto tiempo para recibir un código. Si tiene que esperar mucho o si se pulsa la tecla de espacio volverá, pero sin la bandera de acarreo puesta.

Entrada de la red local: Código de enlace 2Fh, Dirección 6705 (lA31h)

Antes de usar esta rutina se debe abrir y señalar como activo un canal de la red local usando la rutina de Abrir Canal descrita más tarde (ver sección **Salida de la red local**). Esta rutina leerá

un paquete de la red local, a la entrada el registro IX debe señalar el inicio del área de la red local e IX + 11, IX + 12, y IX + 13 y 14 deberán contener los valores correctos para el bloque que se va a recibir (ver detalles sobre la cabecera empleada en la red local en la sección **Salida de la red local**, usando el código de enlace 30H). El número de bloque en IX + 13 y 14 se incrementa después de que cada bloque sea recibido correctamente.

Se pretendía que la bandera de acarreo indicara si se había leído un paquete o si había un error, cuando se devolviese el control a la rutina principal. Sin embargo, la bandera de acarreo puede ser destruida por la modificación del color del borde al salir de la rutina. Se devolverá el control desde ésta cuando un paquete haya sido bien recibido, con la bandera de acarreo bajada. Si el tiempo concedido para que se reciba un paquete ha pasado, se devolverá un error en la suma de comprobación. Alternativamente, si se pulsa BREAK la bandera de acarreo estará levantada.

A causa del problema inherente en esta rutina es más fácil usar la rutina de la ROM de 16K situada en 5606 (15E6h). Recuerde guardar el registro IX antes de la llamada; si no, sería destruido. La bandera de acarreo será puesta a 1 si el registro A no contiene un código recibido.

Salidas

Imprimir en pantalla: Código de enlace 1Ch, Dirección 6636 (19ECh)

Para usar esta rutina el código del carácter debe estar en el registro A antes de llamarla. Esta es un equivalente directo de las rutinas de la ROM de 16K, que colocan la salida hacia la corriente 2 (pantalla principal) e imprimen en la corriente actual usando RST 16; en realidad, usa estas rutinas de la ROM de 16K. Utilizada en conexión con la rutina Leer Tecla (ver sección anterior, Entradas), puede crear una especie de "terminal tonto". Para hacer esto el programa sería:

```
A03D    CF    1    RST     8
A03E    1B    2    DEFB    #1B
A03F    CF    3    RST     8
A040    1C    4    DEFB    #1C
```

El primer RST 8 espera a que una tecla sea pulsada y retorna con su código en el registro A, y el segundo RST 8 imprimirá el carácter en la pantalla principal (o cualquier salida a la que haya sido dirigida la corriente 2).

Enviar a la impresora: Código de enlace 1 Fh, Dirección 6652 (19FCh)

Esta es idéntica a la rutina imprimir en pantalla, ya comentada, excepto que usa la fuente 3 (normalmente la impresora) en lugar de la fuente 2.

Salida al RS 232: Código de enlace 1Dh, Dirección 3162 (C5Ah)

De nuevo, esta rutina se usa poniendo el código que se va a enviar en el registro A, pero utilizando el puerto de salida RS 232 del interfaz. La velocidad de transmisión se toma de la variable del sistema BAUD, y el color del borde, de la variable del sistema IOBORD (ver capítulo 4 para comprender cómo se fijan éstas). No se envía información a no ser que la línea DTR (terminal de datos preparada) esté alta.

El punto de entrada principal llamado por el código de enlace permite que se saque cualquier valor contenido en el registro A; debe tenerse cuidado, por tanto, en evitar enviar códigos de control por equivocación. El otro punto de entrada útil está en 3132 (C3Ch), que comprueba que no se envíen códigos ASCII no imprimibles: cualquiera de los códigos por debajo de 32 (20h) causará un retorno inmediato, excepto el código 13 (0Dh) (retorno de carro), que será enviado seguido por un salto de línea, código 10 (0Ah). Los códigos a partir de 128 (80h) se tratan de

acuerdo con sus tipos. Los caracteres gráficos serán enviados como '?', y las palabras claves expandidas por una llamada a la rutina de la ROM de 16K situada en 3088 (C10h), que necesita que se reste 165 (A5h) al código antes de llamarla.

Salida de la red local

Abrir canal: Código de enlace 2Dh, Dirección 3753 (EA9h)

Antes de que cualquier dato pueda ser enviado o recibido por la red local, se le debe abrir un canal. Esto se hace usando esta llamada después de colocar las variables del sistema D STR1 23766 (5CD6h) y NSTA5 23749 (5CC5h) señalando la dirección de la estación de destino y la de emisión, respectivamente. Se creará un canal de red al final del área CHANS; todos los datos desde el indicado por la variable del sistema PROG 23635/6 (5C53/4h) hasta el indicado por STKEND 23653/4 (5C65/6h) se desplazarán hacia arriba 276 bytes y las variables del sistema relacionadas con esto se actualizarán; todo esto se hace asumiendo que haya espacio suficiente debajo de la dirección indicada en RAMTOP 23730/1 (5CB2/3h). Si no hay espacio suficiente, se producirá un error.

A la salida de la rutina, el registro IX apunta al inicio del nuevo canal. Este es temporal y está indicado como tal porque el bit 7 de IX + 4 está puesto a 1. Para hacerlo permanente, este bit debe ser puesto a 0, y entonces puede ser usado para enviar datos, cargando CURCHL 23633/4 (5C51/2H) con la dirección contenida en el registro IX y usando la rutina RST 16 (10h) de la ROM de 16K para enviar un byte de cada vez.

Después de que el último dato haya sido pasado a la rutina RST 16 (10h), se ha de cerrar el canal usando la rutina indicada por el código de enlace 2Eh (ver **Cerrar canal de la red local**), el cual enviará los datos que queden en la memoria intermedia y la dejará libre. No cerrará ninguna de las fuentes asociadas a este canal, y debe tenerse cuidado para asegurarse de que no se corrompen otros canales y corrientes.

Enviar paquete: Código de enlace 30h, Dirección 3530 (DB2h)

Esta llamada permite que se envíe un "paquete" por la red local. Antes de usarla debe abrir un canal, la cabecera y las variables del sistema inicializadas a sus valores correspondientes y el dato a enviar situado en la memoria intermedia. Para abrir un canal de la red local use el código de enlace 2Dh (ver apartado anterior).

Cuando se llama a esta rutina, primero se envía un 'explorador' que mira si la red local está libre. Entonces se envía la cabecera seguida del bloque de datos. La cabecera está situada al principio del canal de la red local y está apuntada por el registro IX, que señala al primer byte de este canal. Los bytes del IX + 00 a IX + 10 los escribe la rutina de operación del canal, y los bytes IX + 11 al IX + 18 son la cabecera, que tiene los datos siguientes:

IX + 11	La estación receptora
IX + 12	La estación emisora
IX + 13 y 14	El número de bloque
IX + 15	1 si es el último bloque, 0 en otro caso
IX + 16	La longitud de la memoria intermedia (0-255)
IX + 17	Suma de comprobación del bloque de datos
IX + 18	Suma de comprobación de la cabecera

IX + 19 & 20 USADAS SOLAMENTE PARA RECEPCION

IX + 19	La posición del último código tomado de la memoria intermedia

IX + 20 El número de bytes libres en la memoria intermedia
IX + 21 a IX + 275 son los datos que se envían hasta 255 bytes

Las sumas de comprobación las crea la rutina, de modo que no tienen que ser insertadas e IX + 15 se toma del registro A cuando se entra en la rutina, pero todos los demás detalles debe ponerlos el usuario. Además de colocar la cabecera, la variable del sistema D_STR1 23766 (5CD6h) debe señalar a la estación de destino y NTSTAT 23749 (5CC5h) a la emisora. En cada llamada a esta rutina, el número de bloque se incrementará. La dirección de base del canal de la red local es devuelta en el registro IX por la rutina de abrir canal.

Cerrar canal de la red local: Código de enlace 2Eh, Dirección 6692 (lA24h)

Si se llama a esta rutina después de una operación de envío, transmitirá cualquier dato no enviado que quede en la memoria intermedia de la red local, marcándolo como un bloque de fin de fichero; pero después de una operación de recepción descartará cualquier dato que quede en la memoria intermedia.

Los 270 bytes del área de la memoria intermedia serán liberados usando la rutina de la ROM de 16K en 6632 (19E8h) pero dejando la información del canal.

Salida del Microdrive

Abrir canal/abrir fichero: Código de enlace 22h, Dirección 6953 (1B28h)

Antes de que el dato pueda ser enviado al Microdrive, se debe crear un canal específico y un área de mapa para él. El código de enlace 2Bh estaba pensado para este propósito, pero había un error en el programa, aunque este código de enlace puede ser usado para realizar la operación. Primero introduzca en D_STR1 23766/7 (5CD6/7h) el número del drive (1-8); en N_STR1 23770/1 (5CD4/5h), la longitud del nombre del fichero, y en T_STR1 23772/3 (5CD6/7h), la dirección de inicio del nombre del fichero en memoria. La pareja de registros H'L' se debe salvar entonces y, a continuación, se puede llamar a la rutina.

Se abre el canal para lectura si existe el nombre del fichero; si no, se abrirá como un canal de escritura. Para hacer que el canal sea permanente, sus datos deben ser incorporados dentro de la información de las corrientes. La subrutina siguiente lleva a cabo esto:

```
A589  D9        10   EXX                ; salva la dirección de retorno al BASIC
A58A  E5        20   PUSH   HL
A58B  D9        30   EXX
A58C  3A8CA5    40   LD     A, (N)      ; (N) es el número de corriente para el canal
A58F  17        50   RLA                ; lo multiplica por 2
A590  21165C    60   LD     HL, #5C16; dirección base de 1as corrientes
A593  1600      70   LD     D, O        ; guarda el desplazamiento de la corriente S en e1
A595  5F        80   LD     E, A        ; par de registros DE
A596  19        90   ADD    HL, DE      ; ahora HL contiene la dirección de la corriente
A597  E5        100  PUSH   HL          ; lo salva
A598  CF        110  RST    8           ; llama con el código de enlace
A599  22        120  DEFB   #22         ; 22 (abrir canal/fichero)
A59A  E5        130  PUSH   HL          ; a la vuelta de esta rutina HL es el desplazamiento de la corriente
A59B  AF        150  XOR    A           ; la rutina de abrir canal deja el motor
A59C  CF        160  RST    8           ; del microdrive funcionando.
A59D  21        170  DEFB   #21         ; por lo que se tiene que apagar.
A59E  D1        180  POP    DE          ; DE es el desplazamiento de la corriente
A59F  E1        190  POP    HL          ; HL es la dirección de la corriente
A5A0  73        200  LD     (HL), E     ; ajusta la corriente S con e1 desplazamiento correcto
A5A1  23        210  INC    HL
A5A2  72        220  LD     (HL), D
A5A3  DDCB04BE  230  RES 7, (IX+4)      ; al volver de ejecutar el código de enlace 22, IX es el inicio del
                240                     ; área del canal. Poniendo (IX+4) a 0 hace el canal permanente
A5A7  D9        260  EXX                ; recupera la dirección de retorno al BASIC
A5A8  E1        270  POP    HL
```

```
A5A9    D9          280     EXX
A5AA    C9          290     RET             ; fin de la rutina
```

Para escribir o leer desde un Microdrive, el canal creado debe ser activado cargando la variable del sistema CURCHL 23633/4 (5C51/2h) con la dirección base, devuelta en el registro IX después de abrir el canal. Se puede escribir en el cartucho con la rutina RST 16 (10h) de la ROM de 16K o leer con la rutina en 5606 (15E6h) de dicha ROM, que vuelve con el carácter en el registro A. Si se necesita hacer un SAVE, VERIFY, MERGE o LOAD con el Microdrive en lugar de manejarlo como un fichero, se deben realizar los procedimientos explicados más adelante en este capítulo.

Escribir registro: Código de enlace 2611, Dirección 4607 (11FFh)

Este código de enlace escribirá los contenidos de la memoria intermedia del Microdrive al siguiente sector libre en el cartucho. Antes de llamar a esta rutina, la memoria intermedia debe contener la información que se va a salvar; el registro IX debe apuntar al inicio del canal del Microdrive; IX + 11 debe contener la longitud del dato; IX + 13, el número de sector (empezando en 0 para el primer sector de un registro, y automáticamente incrementado cada vez que un registro es enviado o recibido); de IX + 14 a IX + 23 esta guardado el nombre del registro e IX + 25 indica el drive a usar. Las direcciones IX + 82 a IX + 593 están disponibles para almacenar los datos que se van a guardar.

Escribir sector: Código de enlace 2Ah, Dirección 6801 (lA91h)

Este ejecuta la misma acción que el anterior, pero busca un sector de un registro con el número de sector en IX + 13 y, sí existe, lo reescribe con la información que hay en la memoria intermedia. Si el sector no existe, aparecerá un error 'FILE NOT FOUND'. No se comprueba si el sector está libre o no, de modo que se debe comprobar antes que no haya ninguna información importante en ese sector.

Cerrar canal de Microdrive: Código de enlace 23h, Dirección 4777 (12A9h)

Este código de enlace ejecuta la misma acción que el de Cerrar el canal de la red local (código de enlace 2Eh), pero por el Microdrive y, en la entrada, el registro IX debe contener la dirección base del fichero. Si el canal se utilizó para escribir, cualquier información que quede en la memoria intermedia será enviada como mensaje de fin de fichero, y ésta será liberada o, si era un canal de lectura, cualquier cosa que quedase en la memoria intermedia se perderá cuando sea liberado el canal.

Borrar fichero: código de enlace 24h, Dirección 7534 (lD6Eh)

Esta rutina borrará un fichero del Microdrive cuyo nombre esté escrito en memoria y señalada por T_STR1, con la longitud del nombre indicada en N_STR1 en el Microdrive D_STR1.

Entrada desde el Microdrive

Todas las siguientes rutinas de 'lectura' vuelven con el motor del Microdrive funcionando y las interrupciones enmascarables desconectadas.

Leer un registro de caracteres: Código de enlace 27h, Dirección 6679 (lA17h)

Esta rutina leerá un registro de un fichero de caracteres; el número de registro está contenido en IX + 13; el registro IX debe apuntar al inicio del canal del Microdrive; IX + 25 debe contener el

número del drive, y de IX + 14 a IX + 23, el nombre del registro. Suponiendo que el sector del registro esté presente, se leerá dentro de la memoria intermedia y, si no, se producirá un error.

Leer el siguiente registro de caracteres: Código de enlace 25h, Dirección 6665 (lA09h)

Esta es como la anterior, pero cuando se utiliza después de ella, automáticamente leerá el siguiente registro de un fichero, si hay uno, y si no, generará un error.

Leer un registro: Código de enlace 28h, Dirección 6731 (lA4Bh)

Esta rutina leerá un registro del fichero. El registro IX debe apuntar al inicio del canal del Microdrive; IX + 13 debe contener el número de registro (dentro del fichero); IX + 25 debe contener el número del drive, y del IX + 14 a IX + 23, el nombre del fichero. Así, asumiendo que el registro existe, se leerá en la memoria intermedia. Si es un fichero de caracteres, la rutina volverá con la bandera de acarreo bajada y el registro en el área de la memoria intermedia del Microdrive. Si el registro es de un tipo distinto al mencionado, la bandera de acarreo estará levantada, y el registro será borrado de la memoria intermedia.

Leer el siguiente registro: Código de enlace 29h, Dirección 6790 (lA86h)

Esta es como la anterior, pero lee el sector siguiente del fichero. Es similar al código de enlace 25h.

Motor encendido/motor apagado: Código de enlace 21h, Dirección 6135 (17F7h)

El registro A debe ser cargado con 0, para apagar cualquiera de los motores que estén funcionando, o con el número de drive cuyo motor quiere ponerse en marcha. Si el drive no está presente, se producirá un error. Si el drive está presente, el motor será puesto en marcha, y se devolverá el control con las interrupciones desconectadas.

Liberar el canal del Microdrive: Código de enlace 2Ch, Dirección 4292 (10C4h)

El área del canal del Microdrive señalada por el registro IX será liberada.

Cualquier corriente que esté usando el canal será cerrada y, el área de mapa del Microdrive, será también liberada si no la usa otro canal. Toda la memoria situada encima del canal liberado hasta RAMTOP se moverá hacia abajo 627 bytes.

Mirar teclado: Código de enlace 20h, Dirección 6657 (lA0lh)

Esta rutina vuelve con la bandera de acarreo levantada si se está pulsando una tecla.

Insertar variables: Código de enlace 3lh, Dirección 6568 (l9A8h)

Esta rutina inserta las variables del sistema extra que necesita el Interfaz 1.

En realidad, es una simple instrucción RET, ya que las variables se crean automáticamente cuando se produce el primer error de 16K.

ROM 2: Código de enlace 32h, Dirección 6564 (l9A4h)

Quizás este sea el más útil de todos los códigos de enlace, ya que permite que la ROM 8K sea paginada a voluntad; por tanto, se puede llamar a cualquier rutina. Para usarlo, la dirección de la rutina a la que quiere acceder debe colocarse en la variable del sistema HD_11. El código de enlace puede ser usado a continuación. Desafortunadamente, sólo se puede pasar directamente con este código de enlace el registro A, pero esto no es un problema realmente.

Cuando se usa el código de enlace, la dirección en HD_11 23789/90 (5CED/Eh) es puesta en la pila de la máquina seguida por la dirección de retorno para conmutar desde la ROM de 8K a la de 16K (1792 (700h)). Si usted hace que la dirección en HD_11 señale a su propio programa, entonces puede sacar las dos direcciones de retorno de la pila con la ROM de 8K presente bajo el control de su programa. Cualquiera de los registros puede ser usado ahora libremente con las rutinas de esta ROM. Para volver a la ROM de 16K, todo lo que usted necesita hacer es llamar (CALL) a la dirección 1792 (700h), y la ROM de 8K se desconectará.

Nota: Algunas rutinas comprueban la bandera "sintax/run" en FLAGS 23611 (5C3 Bh) IY + 1, y producen un retorno al intérprete BASIC por sintaxis.

Algunas de las rutinas del Microdrive también vuelven al BASIC mediante la dirección en ERR SP 23613/4 (5C3D/Eh), si no ha habido error, o por medio de un error de la ROM de 8K, si lo ha habido. El error de 16K puede ser modificado para que apunte a la rutina que ha realizado la llamada, alterando la dirección en ERR SP de modo que apunte a la dirección de retorno de su rutina almacenada en la pila de la máquina. Esto se demuestra en el programa DeBASE en el apéndice G, en las rutinas SALVAR/CARGAR del Microdrive.

Las rutinas siguientes se llaman con el código de enlace RST 8 DEFB 32h, poniendo la dirección en HD_ll.

Catálogo del cartucho: Código de enlace 32h, Dirección 7256 (lC58h)

Antes de usarla, salve los registros H'L'. D_STR1 debe contener el número del Microdrive y S_STR1 el número de la corriente donde se debe imprimir el catálogo. Entonces se realiza la llamada, y después de hacer el catálogo hay que devolver a los registros H'L' sus valores.

Formatear cartucho: Código de enlace 32h, Dirección 7022 (lB6Eh)

Antes de usar esta rutina hay que salvar los registros H'L'. N_STR1 debe contener la longitud del nombre que se va a dar al cartucho y T_STR1 la dirección del primer carácter del nombre. Los registros H'L' han de ser restaurados cuando se termine la ejecución para poder volver al BASIC.

Ejecución: Código de enlace 32h, Dirección 2709 (A95h)

Esta es la más simple de las rutinas. Carga un programa llamado 'Run' del Microdrive 1. Todo lo que hace falta es hacer un salto a la rutina en ROM. Esto se puede hacer después de que la ROM de 16K haya sido desconectada, como se indicó en la sección anterior (Código de enlace 32h).

Las siguientes rutinas (SAVE, LOAD, VERIFY y MERGE) requieren que se construya una cabecera antes de ser usadas. Los detalles de la composición de un canal de Microdrive se dan en el manual del Microdrive e Interfaz 1 (ver páginas 47 y 48) de modo que no repetiré estos datos aquí. Sin embargo, explicaré cómo se hace una cabecera, de forma que el Microdrive se pueda usar desde el lenguaje máquina.

Construyendo una cabecera de Microdrive

La cabecera del Microdrive se compone de un modo parecido a la cabecera de la cinta, descrito en el capítulo 2, pero no es necesario construir la cabecera en el canal del Microdrive, ya que hay una rutina en ROM que lo hará por usted. Todo lo que hay que hacer es colocar las variables del sistema correctamente. La subrutina para hacer esto dentro de su programa sería algo así:

```
A86E    21E65C           LD     HL, 23782      ; HD_00
A871    3600             LD     (HL), T        ; el dato que se escribe debe ser 0
                                               ; para programa, 1 para matriz
                                               ; numérica, 2 para matriz alfanumérica
                                               ; y 3 para código binario
A873    23               INC    HL
A874    1174A8           LD     DE, LEN        ; longitud del bloque de datos
A877    73               LD     (HL), E
A878    23               INC    HL
A879    72               LD     (HL), D
A87A    23               INC    HL
A87B    117BA8           LD     DE, INICIO     ; dirección de inicio del
                                               ; área donde se leen o
A87E    73               LD     (HL), E        ; escriben los datos,
                                               ; si es cero se usa la información
A87F    23               INC    HL             ; del cartucho.
A880    72               LD     (HL), D
                                               ; El resto de la rutina se tiene que ejecutar solamente
                                               ; si se manejan programas BASIC y matrices. Si no se
                                               ; salta a EJECUT
A881    23               INC    HL
A882    1182A8           LD     DE, DATO       ; longitud del programa
                                               ; o nombre de variable
A885    73               LD     (HL), E        ; tal como se describe en el capitulo
A886    23               INC    HL             ; 2 en la parte de salvar y cargar.
A887    72               LD     (HL), D
A888    23               INC    HL
A889    1189A8           LD     DE, AUTO       ; numero de la línea de arranque
A88C    73               LD     (HL), E        ; automático, si hay alguno. Sino un
A88D    23               INC    HL             ; número superior a diez mil.
A88E    72               LD     (HL), D
A88F    21D95C   EJECUT  LD     HL, 23769      ; indicamos que es una operación
A892    364D             LD     (HL), "M"      ; de Microdrive (la letra en mayúscula)
A894 FDCB7CC6            SET    BIT, (IY+124)  ; bit 4 para cargar, 5 salvar,
                                               ; 6 unir merge) y 7 verificar.
A898    2198A8           LD     HL, NLEN       ; longitud del nombre del fichero
                                               ; máximo diez caracteres o códigos especiales.
A89B    22DA5C           LD     (23770), HL    ; esta es NSTR1
A89E    21B4A8           LD     HL, NOMBRE     ; dirección de inicio del nombre.
A8A1    22DC5C           LD     (23772), HL    ; T_STR1
A8A4    21A4A8           LD     HL, DRIVE      ; el número de la unidad (1-8)
A8A7    22D65C           LD     (23766), HL    ; D_STR1
A8AA FDCB7C6E            BIT    5, (IY+124)    ; comprueba si es una operación SAVE
A8AE    C27F1E           JP     NZ, #1E7F      ; si es, va a la rutina
                                               ; correspondiente en ROM
A8B1    C3AF08           JP     #08AF          ; si no va a la de cargar/verificar/unir.
A8B4             NOMBRE  DEFS   10             ; l0 bytes libres para el nombre
0000             T       EQU    0
0000             BIT     EQU    0
```

La rutina anterior debe llamarse desde el programa principal. Después de que la operación ha sido ejecutada, el retorno (RET) se produce, desde la rutina en ROM a la que ha realizado la llamada, mediante la dirección en la pila señalada por ERR SP (ver el programa DeBASE en el apéndice G para obtener más detalles sobre este tema). La ROM de 16K será conectada por las rutinas SAVE o LOAD.

Cualquier error cuando se construya la cabecera causará o bien un error del BASIC o bien un bloqueo de la máquina, así que se debe tener mucho cuidado. Como con la mayoría de las rutinas de la ROM de 8K, es importante salvar los registros H'L' antes de usar las rutinas y recuperarlos después.

Cuando se está escribiendo un programa de código máquina usando las rutinas de la ROM, es muy importante recordar qué ROM estará conectada en un momento dado. No sólo todas las rutinas son diferentes, sino que, además, todos los RST en la ROM de 8K son diferentes de los de la de 16K.

Es posible cambiar libremente entre las dos ROMs usando el código de enlace 32h desde la ROM de 16K y RST 10h desde la ROM de 8K. El uso del código de enlace 32h ha sido explicado ya, pero los detalles de cómo usar RST 10h y los otros RST en la ROM de 8K se dan a continuación:

```
A0E4 E1              RST_O   POP HL            ; quita la dirección de retorno de la
                                               ; pila de la máquina.
A0E5 FD367C00                LD (IY+124), 00   ; esta es la variable FLAGS 3
A0E9 C30007                  JP #700           ;  la dirección de retorno a la ROM de 16 K
```

La dirección en la parte superior de la pila en esta fase ha de ser el retorno a la rutina que originalmente llamó a la ROM de 8K.

```
0008                         ORG 8
0008   2A0800        RST_8   LD HL, (CH_ADD)   ; esta no hace nada cuando se la
                                               ; llama desde la ROM de 16 K
000B   El                    POP HL            ; la dirección del código de error
000C   E5                    PUSH HL           ; lo salva de nuevo
000D   C39AOO                JP #009ª          ; esto comprobara que orden se esta
                                               ; ejecutando: volver a la rutina de errores de la ROM de 16 K,
                                               ; ejecutar un código de enlace o abrir un canal.
```

Esto no tiene utilidad práctica cuando se conecta la ROM de 8K.

La RST 16 (10h) en la ROM de 8K llama a una rutina de la ROM de 16K, la dirección de la cual está contenida en una palabra (DEFW) después de la instrucción RST. Después de que la rutina de 16K haya sido ejecutada, el control se devuelve a la dirección posterior a DEFW en la rutina que realizó la llamada.

```
0010                         ORG 16
0010   22BA5C        RST_16  LD (#5CBA), HL    ; salva HL para recuperarla al terminar.
0013   E1                    POP HL            ; almacena en HL la dirección de la palabra
0014   D5                    PUSH DE           ; salva DE en la pila de la máquina.
0015   186A                  JR #81
                                               ; 0081h
0017   5E                    LD E, (HL)        ; pone la dirección de la rutina de la
0018   23                    INC HL            ; ROM de 16K que se quiere llamar en DE
0019   56                    LD D, (HL)
001A   23                    INC HL            ; HL contiene la dirección de retorno
                                               ; (la posición que va después de DEFW)
001B   E3                    EX (SP), HL       ; hace que esta sea la nueva dirección de retorno.
001C   EB                    EX DE, HL
001D   210000                LD HL,0           ; esto hace que la rutina RST_8 sepa si
0020   E5                    PUSH HL           ; es un retorno de la ROM de 8K o de 16K.
0021   210800                LD HL,8           ; pone la dirección de la rutina de
0024   E5                    PUSH HL           ; errores en la pila para volver luego.
0025   21B95C                LD HL, #5CB9      ; la dirección de la subrutina
0028   E5                    PUSH HL           ; la guarda en la pila para volver.
0029   C30007                JP #0700          ; esta es una dirección de retorno que conecta
                                               ; la ROM de 16 K y vuelve
                                               ; 0700h
002C C9                      RET               ; fin del proceso.
```

He dado este listado completo porque es un método interesante para transferir el control.

Cuando se hace un retorno a 0700h, conectando la ROM de 16K, la dirección que se coge de la pila es 5CB9h. Esta dirección contiene 21h (la instrucción LD HL, NN). Como puede verse en

el listado, el NN era el valor en HL cuando se ejecutó RST. La dirección siguiente, 5CBCh, contiene CDh (la instrucción CALL NN). De nuevo esta dirección fue cargada antes del DEFW después del RST 16 (10h). El retorno desde la rutina de 16K será a 5CBFh, que contiene 22h (la instrucción LD (NN), HL). La NN en este caso es 5CBAh, de modo que el nuevo valor del registro HL es salvado donde el original. Finalmente, está la instrucción RET. Esta será un retorno a la dirección en la parte superior de la pila de la máquina, ahora 8 (es decir, la rutina de error RST). Esta vez cuando la dirección para la DEFB sea sacada de la pila será 0. Esto le dirá a la ROM de 8K que el registro HL se tiene que cargar desde (5CBAh), y después de esto hay todavía otro retorno (RET), esta vez a la dirección después de los dos DEFB a continuación del RST 16 (10h).

La siguiente RST puede utilizarse para generar los mensajes de error de la ROM suplementaria, del mismo modo que la RST 8 lo hace en la ROM principal:

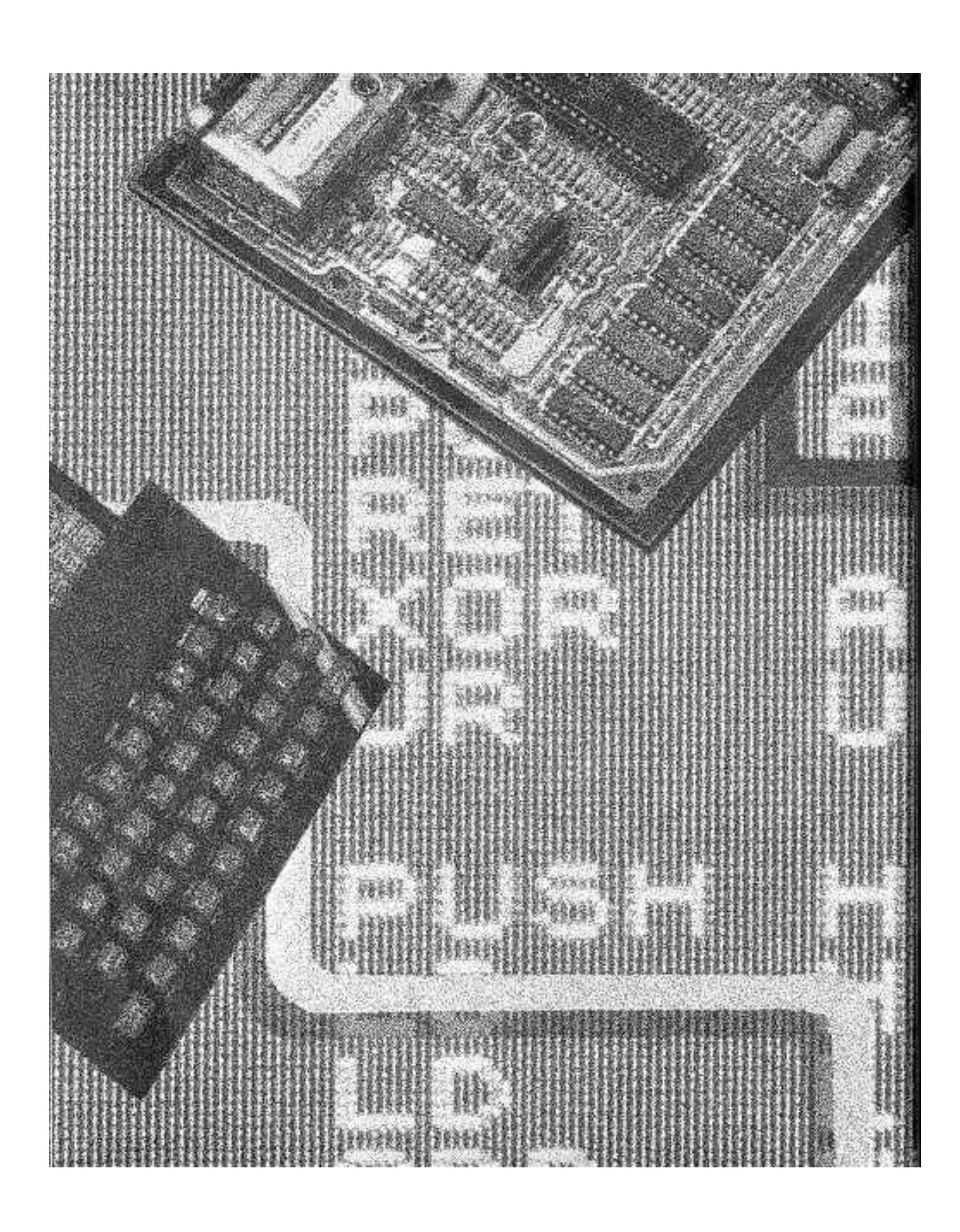
```
0018               5                ORG 24
0018 FDCB017E      10     RST_24    BIT 7, (IY+1)      ; esta es la variable FLAGS
001C      C9       20               RET                ; la bandera cero estará puesta si se esta
                   30                                  ; comprobando la sintaxis de una línea BASIC
001D      DF       40     RST_32    RST 24             ; mira si se está comprobando la
                   50                                  ; sintaxis o se está ejecutando un programa.
001E      2848     60               JR Z, #68          ; esto modificara el puntero de la pila (SP)
                   70                                  ; para que indique hacia (ERR_SP), meterá en X_PTR el
                   80                                  ; valor de CH_ADD y llamara a la rutina de entrada
                   90                                  ; de líneas de la ROM por medio de la rutina de
                   100                                 ; borrar el calculador e indicará la posición del error
0020      1818     110              JR #3A             ; esto produce el mensaje de error.
                   120                                 ; el código de un error de la ROM del
                   130                                 ; interfaz se coge del byte
                   140                                 ; siguiente al RST.
```

Esta rutina (RST) se utiliza para generar un error en la ROM de 16K, cargando ERR-NR con el código del error antes de efectuar la llamada.

```
0028               5                ORG 40
0028 FDCBO2EE      30     RST_40    SET 5, (IY+2)      ; esta es la variable TV FLAG
002C      1812     40               JR #40             ; aquí se comprueba si se ha producido un
                   50                                  ; error durante la comprobación de sintaxis o al
                   60                                  ; ejecutar. Si es durante la ejecución, se irá a la
                   70                                  ; rutina de errores de la ROM de 16 K. Si no se saltara a 0068h
                   80                                  ; como se ha explicado antes.
002E      C3F701   90     RST_48    JP #01F7           ; mira si las variables del interfaz
                   100                                 ; 1 están presentes y si no lo están, las inserta.
0031      FB       110    RST_56    EI                 ; permite las interrupciones enmascarables
                   120                           ; que son las que miran el teclado si la ROM de 16K está conectada.
0032      C9       140              RET                ; Tenga en cuenta que esta rutina
                   150                                 ; no explora el teclado.
```

La rutina de interrupción no enmascarable en 102 (66h) consiste en una sencilla instrucción RETN.

Se explicarán más rutinas de la ROM de 8K en el capítulo 7, pero hay pocas que sean útiles para otros propósitos. Sus usos y direcciones han sido detallados ya en la sección de códigos de enlace.


PUSH

4. Las variables del sistema

El intérprete de BASIC necesita algunos datos para estar al corriente de dónde se almacena la información y qué se supone que está haciendo. Dado que está guardado en ROM (memoria de sólo lectura) tiene que usar RAM (memoria de acceso aleatorio) para almacenar esta información. Como la ROM no se puede alterar, las direcciones de las variables del sistema tienen que estar definidas previamente, de forma que se pueda acceder a ellas fácilmente. En el Spectrum esto se consigue por dos métodos: teniendo fijadas sus direcciones en un área específica de la memoria e indexándolas con el registro IY, que señala siempre a 5C3Ah (ERR NR).

Variables del sistema de 16K

KSTATE: Direcciones 23552-23559 (5C00h-5C07h)

Estas direcciones se usan para leer el teclado. Se dividen en dos grupos de cuatro y cada grupo es tratado igual que el otro. El uso de un grupo dependerá del estado del otro.

El valor en mayúsculas de una tecla que se está pulsando es almacenado en la primera dirección de cada grupo o, si no hay ninguna tecla pulsada, se almacena 255 (FFh), que señala que el grupo se puede usar. La cuenta para que un conjunto esté libre se almacena en el segundo byte, libre cuando sea 0 e, inicialmente, colocada a 5. El retardo de repetición está en el tercer byte, cargado inicialmente desde REPPER. El código ASCII de la tecla pulsada, cuando está en uso un grupo, está en la última dirección de este grupo hasta que el contador que indica que el grupo está libre llega a 0. Para comprender el sentido de esto explicaré la rutina que usa estas variables.

Si se pulsa una tecla, la subrutina de exploración mira para ver si el primer grupo está vacío. Si lo está, se usará entonces este grupo; si no, mirará en el segundo. Cuando un grupo está libre, el código CAPS SHIFT de la tecla que está pulsada se salva en la primera dirección. En la segunda dirección se almacena un 5 (el contador que indica que el grupo está libre, el cual sirve también para evitar los rebotes del teclado); la tercera dirección se carga con el retardo de repetición. El código ASCII de la tecla pulsada es decodificado, salvado en la última dirección y copiado dentro de LAST K, el bit 5 de FLAGS se coloca a 1 para señalar que una nueva tecla ha sido pulsada y se devuelve el control a la rutina que la ha llamado. Cuando ningún grupo esté libre, el contador en el segundo byte de cada grupo es decrementado, y se devuelve el control a la rutina que ha efectuado la llamada.

Una vez que uno de los contadores alcanza 0, el código de la tecla pulsada (cualquiera de ellas) se salva y se hace una comparación entre el código de las teclas en los dos grupos. Si los códigos son iguales, se está pulsando una tecla y, entonces, la rutina permite la repetición de la tecla. Esta no es la rutina de la tecla de eliminación de rebotes, pero sí la rutina de repetición de tecla. El retardo de repetición está contenido en el tercer byte y cargado inicialmente desde la variable del sistema REPDEL.

Una vez que el retraso para la primera repetición ha pasado (la rutina debe ser llamada el número de veces indicado por el contador de retardo), si el código de las teclas ha coincidido de forma continua, entonces el tercer byte se carga con el valor de la variable del sistema REPDEL; se pasa de nuevo el código a la variable del sistema LAST KZ, el bit 5 del grupo FLAGS es vuelto a poner a 1 y se realiza un retorno a la rutina que hizo la llamada. En llamadas sucesivas a la rutina de exploración del teclado el proceso completo se repite, pero con el retraso menor leído de REPPER, hasta que los códigos de los dos grupos no coinciden. Si las teclas no coinciden, entonces el tratamiento es similar que el dado a un grupo libre.

El resultado neto de todo esto es que se admiten hasta dos teclas pulsadas en rápida sucesión, permitiendo que se pulse una segunda tecla cuando toda vía no ha pasado el retardo de la anterior, evitando la pérdida de la segunda si no está apretada el suficiente tiempo como para que el primer conjunto quede libre.

LAST K: Dirección 23560 IY-50 (5C08h)

Aquí se guarda el código de la última tecla pulsada.

REPDEL: Dirección 23561 IY-49 (5C09h)

En ésta se guarda el número de veces que la rutina de exploración del teclado debe ser llamada antes de que una tecla se repita.

REPPER: Dirección 23562 IY-48 (5C0Ah)

Esta contiene el número de veces que la rutina de exploración del teclado debe ser llamada entre repeticiones sucesivas de una tecla.

DEFADD: Direcciones 23563/4 IY-47 (5COB/Ch)

Aquí se tiene o la dirección de una función definida por el usuario que está siendo evaluada ó 0.

K DATA: Dirección 23565 IY-45 (5C0Dh)

Se usa como almacenamiento temporal de la información del color cuando se están introduciendo controles de color (es decir, el segundo byte, el número de color, después de shift E).

TVDATA: Dirección 23566/7 IY-44 (5C0E/Eh)

Es como la anterior, pero para la salida y se usa con AT y TAB. Esta es la razón de que sean dos bytes.

STRMS: Direcciones 23568-23605 IY-42 (5C10h-5C35h)

Contienen las direcciones de los canales unidos a las corrientes. Inicialmente, las corrientes - 3 a 3 están en los primeros 14 bytes, pero la información de corrientes extras (hasta un total de 19) se añade, cuando se abren, en el sitio pertinente.

CHARS: Direcciones 23606/7 IY - 4 (5C36/7h)

Contiene la dirección del inicio teórico del grupo de caracteres, cada carácter consta de ocho bytes; en realidad, esta dirección es 256 bytes menor que el inicio real, ya que todos los caracteres están direccionados con respecto a esta posición, multiplicando su código ASCII por ocho. Además, los caracteres ASCII del 0 al 31 no son imprimibles y son manejados de forma diferente, no con esta tabla.

El último código válido ASCII es 127 (7Fh), pero el Spectrum usa los 128 valores restantes de ocho bits para códigos especiales. Estos se manejan aparte y no forman parte de este conjunto.

RASP: Dirección 23608 IY - 2 (5C38h)

Esta contiene la longitud del zumbido generado cuando una línea del BASIC excede de 22 líneas o no queda memoria disponible.

PIP: Dirección 23609 IY - 1 (5C39h)

Contiene la duración del tono producido cuando se pulsa una tecla.

ERR NR: Dirección 23610 IY + 0 (5C3Ah)

Esta contiene uno menos que el código que se genera en un error. Puede usarse para generar sus propios mensajes o para usar los ya existentes para sus propios fines (para más detalles, ver el capítulo 2).

FLAGS: Dirección 23611 IY + 1 (5C3Bh)

El bit 0 se pone a 1 para suprimir el espacio anterior cuando se imprime una palabra clave si ya existe un espacio, y se pone a 0 si se tiene que imprimir un espacio.

El bit 1 se pone a 1 cuando la corriente 3 (normalmente la impresora) va a usarse como salida de la rutina RST 16, pero se pone a 0 para la fuente 2 (normalmente, la pantalla principal).

El bit 2 se pone a 1 cuando la impresión se va a realizar con el modo L, y se pone a 0 para el modo K.

El bit 3 se coloca a 1 para señalar el modo L, normalmente cuando se toma una entrada, pero se coloca a 0 para señalar al modo K.

El bit 5 se coloca a 1 si se ha pulsado una nueva tecla desde que se puso a 0 la última vez (más detalles se pueden encontrar en "Explorando el teclado", en el capítulo 2, y en la variable del sistema KSTATE).

El bit 6 se usa para indicar si una expresión es numérica o es alfanumérica, poniéndose a 1 para numérica ya 0 para una alfanumérica; se usa por el intérprete de BASIC.

El bit 7 se coloca a 0 cuando el intérprete de BASIC está comprobando la sintaxis de una línea introducida ya 1 cuando el programa está siendo ejecutado.

TV FLAG: Dirección 23612 IY + 2 (5C3Ch)

El bit 0 se coloca a 1 si se está manejando la pantalla inferior ya 0 para la pantalla principal.

El bit 3 señala que el modo actual puede haber cambiado y necesita ser verificado.

El bit 4 se pone a 1 cuando se está imprimiendo un listado automático, si no se pone a 0.

El bit 5 señala que la pantalla inferior necesita borrarse (por ejemplo: cuando se va a imprimir un informe).

ERR SP: Direcciones 23613/4 IY + 3 (5C3D/Eh)

Estas posiciones contienen la dirección en la pila de la máquina del retorno que se usa en el caso de que se produzca un error en el BASIC cuando está ejecutando una orden. Lo usa la rutina de manejo de errores y es frecuentemente cambiado por el BASIC. Se puede cambiar desde el código máquina para hacer que un error vaya a su propia rutina, como se demuestra en el programa DeBASE en el apéndice G, en la sección Salvar/cargar del Microdrive. Si ocurriese un error, éste debe ser puesto a 0, y el error cancelado haciendo que ERR NR valga 255 (FFh). Nótese que 0 OK se considera un error.

LIST SP: Direcciones 23615/6 IY + 5 (5C3F/40h)

LIST SP se usa para salvar la dirección del puntero de la pila (stack pointer), de forma que se pueda recuperar después de que se complete un listado automático. Esto es necesario porque el listado se puede terminar en sitios diferentes, con valores diferentes en la pila de la máquina (por ejemplo: por una ‘n’ en respuesta a un mensaje ‘Scroll?').

MODE: Dirección 23617 IY + 7 (5C41h)

Determina qué cursor se usa en la entrada, pero sólo afectará a la primera tecla pulsada, excepto cuando se cambia para forzar el modo de gráficos. Sin embargo, puede ser conveniente conseguir diferentes cursores para la entrada (por ejemplo: con POKE MODE,32 conseguirá una entrada de gráficos parpadeantes). La experimentación es el mejor camino para saber cómo se usa esta variable, ya que el sistema no se bloquea.

NEWPPC: Direcciones 23618/9 IY + 8 (5C42/3h)

Esta contiene el número de línea de la siguiente sentencia que se va a interpretar.

NSPPC: Dirección 23620 IY + 10 (5C44h)

Este es el número de la sentencia dentro de la línea que se va a ejecutar a continuación. Haciendo un poke en el número de línea, y después en el número de la sentencia, fuerza un salto a este sitio durante la ejecución.

PPC: Direcciones 23621/2 IY + 11 (5C45/6h)

Contiene el número de línea de la sentencia actual que se está ejecutando. También es usada para ejecutar automática mente un programa tomando el dato de la cabecera.

SUBPPC: Dirección 23623 IY + 13 (5C47h)

Esta contiene el número de sentencia que se está ejecutando actualmente.

BORDCR: Dirección 23624 IY + 14 (5C48h)

Contiene el valor del color del borde * 8 más los atributos de la pantalla inferior. Los bits 7 y 6 se usan para hacer que la pantalla inferior parpadee o esté brillante.

E PPC: Direcciones 23625/6 IY + 15 (5C49/ Ah)

E PPC contiene el número de la línea actual (es decir, la línea que será señalada con el cursor y llevada dentro del área de edición señalada por E LINE en el comando EDIT).

VARS: Direcciones 23627/8 IY + 17 (5C4B/Ch)

Estas contienen la dirección de inicio del área de variables. Cuando se use en conjunción con E LINE se puede calcular el tamaño total de las variables del programa BASIC.

DEST: Direcciones 23629/30 IY + 19 (5C4D/Eh)

Estas contienen la dirección de la primera letra del nombre de la variable actualmente en uso. Si una nueva variable va a ser usada, entonces vale 80h, que es la marca de fin situada inmediatamente antes de la dirección en E LINE, que es donde se coloca una nueva variable.

CHANS: Direcciones 23631/2 IY + 21 (5C4F/50h)

Contienen la dirección de inicio del área de información del canal en la cual se guardan los detalles de los canales abiertos.

CURCHL: Direcciones 23633/4 IY + 23 (5C51/2h)

Aquí se almacena la dirección del inicio del área de información del canal actualmente en uso.

PROG: Direcciones 23635/6 IY + 25 (5C53/4h)

Esta es la dirección del inicio del área del programa BASIC, que será el byte siguiente al área de información del canal ya cualquier memoria intermedia de entrada/salida usada por el Interfaz 1.

NXTLIN: Direcciones 23637/8 IY + 27 (5C55/6h)

Esta es la dirección de inicio de la siguiente línea del BASIC en un programa.

DATADD: Direcciones 23639/40 IY + 29 (5C57/8h)

Es la dirección donde termina el último elemento usado en un DATA, o el inicio de la línea asignada por un comando RESTORE, o la primera después de ésta si no existiese. De este modo, se está al corriente de cuál es el siguiente dato a usar y, si no hay más después de este indicador, se producirá un error out of data.

E LINE: Direcciones 23641/2 IY + 31 (5C59/Ah)

E LINE es la dirección de inicio del área de edición, que también será el inicio de cualquier línea que se esté editando.

K CUR: Direcciones 23643/4 IY + 33 (5C5B/Ch)

Es la dirección del cursor dentro de la línea. Se utiliza para editar o crear una nueva línea en el área de edición.

CH ADD: Direcciones 23645/6 IY + 35 (5C5D/EH)

Esta es la dirección del carácter siguiente que se va a interpretar por el BASIC, (Nota: Cualquier dato numérico está marcado con un CHR$ 14, y es ignorado.)

XPTR: Direcciones 23647/8 IY + 37 (5C5F/60h)

Estos bytes almacenan la dirección en la que la sintaxis de una línea BASIC ha fallado cuando ha sido introducida, marcada con '?'. Se usa también para almacenar información temporal por el intérprete.

WORKSP: Direcciones 23649/50 IY + 39 (5C61/2h)

Contiene la dirección del área de trabajo temporal, el primer byte de espacio libre creado al hacer un CALL 5717 (1655h) (ver capítulo 5; para más detalles, sección Corrientes estándar).

STKBOT: Direcciones 23651/2 IY + 41 (5C63/4h)

Esta es la dirección de la base (inicio) de la pila del calculador. La explicación completa la encontrará en el capítulo 8.

STKEND: Direcciones 23653/4 IY + 43 (5C65/6h)

Esta es la dirección de la parte superior (final) de la pila del calculador. También en el capítulo 8 encontrará la explicación completa.

BREG: Dirección 23655 IY + 45 (5C67h)

Este es el registro B para uso del calculador (ver capítulo 8, para ampliar detalles).

MEM: Dirección 23656/7 IY + 46 (5C68/9h)

MEM proporciona la dirección del área de memoria del calculador (para más detalles, ver capítulo 8).

FLAGS2: Dirección 23658 IY + 48 (5C6Ah)

El bit 0 se pone a 1 si no es necesario borrar la pantalla principal cuando se está introduciendo una línea dentro del área de edición.

El bit 1 se pone a 1 si la memoria intermedia de la impresora ha sido usada por una rutina de la ROM de 16K, y se pone a 0 después de que sea borrada por la rutina Borrando la memoria intermedia de la impresora descrita en el capítulo 2.

El bit 2 puesto a 1 indica que la pantalla está limpia.

El bit 3 puesto a 1 indica que el teclado está en mayúsculas.

El bit 4 puesto a 1 indica que el canal K está en uso.

DF SZ: Dirección 23659 IY + 49 (5C6Bh)

Este es el número de líneas asignadas a la pantalla inferior. Puede causar el bloqueo del ordenador si se inicializa con un número inferior a 3, dado que siempre se deja una línea libre entre la pantalla principal y cualquiera de los mensajes. Si no hay sitio para imprimirlos, el intérprete se bloquea.

STOP: Direcciones 23660/1 IY + 50 (5C6C/Dh)

Contiene el número de la primera línea del programa para ser listada para una instrucción LIST del BASIC.

OLDPPC: Direcciones 23662/3 IY + 52 (5C6E/Fh)

Esta dirección almacena el número de línea que contiene la sentencia detrás de la cual se ejecutó un BREAK o un STOP (es decir, la línea que se interpretará después de una instrucción CONTINUE).

OSPCC: Dirección 23664 IY + 54 (5C70h)

Esta es igual que la anterior, pero almacena el número de sentencia dentro de la línea.

FLAGX: Dirección 23665 IY + 55 (5C71h)

La variable FLAGX es un equivalente tosco de la variable del sistema FLAGS (Dirección 23611 (5C3Ch), y lo usa el intérprete en su lugar cuando se está ejecutando una instrucción INPUT.

El bit 1 se pone a 1 si el intérprete BASIC maneja una nueva variable.

El bit 5 se pone a 1 cuando la ROM 16K está en modo de entrada, y se reinicializa cuando está en modo de edición.

El bit 6 se pone a 1 si se maneja una cadena de caracteres.

El bit 7 se pone a 1 si el intérprete BASIC está tratando con una instrucción INPUT LINE.

STRLEN: Direcciones 23666/7 IY + 56 (5C72/3h)

Estas direcciones contienen cada una la longitud de una variable alfanumérica existente que está siendo usada actualmente; o en una numérica o una variable alfanumérica nueva, el byte bajo contendrá la letra de la variable (en la forma descrita en la sección Salvando y cargando, ver capítulo 2).

T ADDR: Direcciones 23668/9 IY + 58 (5C74/5h)

Estas posiciones contienen la dirección del elemento siguiente en las tablas de sintaxis localizadas desde 6728 (lA48h) en la ROM principal. Sin embargo, es usada también para otros fines por algunas rutinas.

SEDD: Direcciones 23670/1 IY + 60 (5C76/7h)

Estas son la base para un número aleatorio. Se toma desde los dos bytes bajos de FRAMES, si una instrucción RANDOMIZE no tiene número; si no, se toma el número de RANDOMIZE.

FRAMES: Posiciones de memoria 23672-23674 IY + 62 (5C78h-5C7Ah)

En esta dirección se encuentra un contador compuesto por tres bytes. Se inicializa a 0 cuando el Spectrum se enciende, y se incrementa cada vez que se llama a la rutina de interrupción normal. El byte menos significativo está en 23672 (5C78h).

UDG: Posiciones 23675/6 IY + 65 (5C7B/Ch)

Esta es la dirección del primer gráfico definido por el usuario.

COORDS: Posiciones 23677/8 IY + 67 (5C7D/Eh)

COORDS guarda las coordenadas X e Y del último punto dibujado. Se puede usar para cambiar la posición inicial de DRAW o CIRCLE sin tener que dibujar el inicio; X (horizontal) está en 23677 (5C7Dh).

P POSN: Posiciones 23679 IY + 69 (5C7Fh)

Esta contiene el número de la columna de la posición siguiente a usar en la memoria intermedia de la impresora (como **S POSN: Posiciones 23688/9 IY + 78 (5C88/9h)** explicada más adelante, pero por impresora).

PR CC: Posiciones 23680 IY + 70 (5C80h)

Este es el byte bajo de la dirección actual de la memoria intermedia de la impresora. En efecto, es idéntico a **DF CC: Posiciones 23684/5 IY + 48 (5C84/5h)**, y la memoria de la impresora se puede mover cambiando el llamado byte no usado que viene a continuación, que es, en realidad, el byte alto de la variable PR CC. Desafortunadamente, éste se reinicializa al final de cada línea impresa para señalar a la dirección original. Atención, por tanto, si se ha movido.

NO USADA: Posición 23681 IY + 71 (5C81h)

En efecto, ésta no se usa si la memoria intermedia de la impresora está en el sitio normal (ver **PR CC: Posición 23680 IY + 70 (5C80h)**, pero no está libre para usar, excepto como se ha dicho anteriormente.

ECHO E: Posiciones 23682 IY + 72 (5C82/4h)

Como **S POSN: Posiciones 23688/9 IY + 78 (5C88/9h)**, ésta es la columna y el número de línea para la siguiente posición impresa; pero en este caso, para la memoria intermedia de salida. Se usa cuando entra una línea de BASIC.

DF CC: Posición 23684/5 IY + 74 (5C84/5h)

Contiene la dirección de la línea superior de pixels en la siguiente posición a imprimir. DF CC se puede usar para cambiar la posición de los caracteres normales impresos, modificándola para la línea inferior, pero puede causar efectos inesperados, dado que la rutina RST 16 (10h) sólo añade 256 (100h) para cada línea subsiguiente de puntos de un dato a imprimir. El apéndice C, **El mapa de pantalla del Spectrum**, se puede usar para ver el efecto producido cuando se baja más de una línea de puntos.

DFCCL: Posiciones 23686/7 IY + 76 (5C86/7h)

Esta es la versión para la pantalla inferior de la rutina **DFCC: Posiciones 23684/5 IY + 74 (5C84/5h)**.

S POSN: Posiciones 23688/9 IY + 78 (5C88/9h)

Estas posiciones contienen la columna y línea de la siguiente posición de impresión en la pantalla principal, y son actualizadas por la rutina CALL 3545(DD9h), explicada en el capítulo 2 (donde 33 es la columna izquierda, y 24 es la línea de la parte superior).

SPOSNL: Posiciones 23690/1 IY + 80 (5C8A/Bh)

Como **S POSN: Posiciones 23688/9 IY + 78 (5C88/9h)**, estas posiciones contienen la columna y el número de línea de la pantalla inferior.

SCR CT: Posición 23692 IY + 82 (5C8Ch)

Esta posee uno más que el número de veces que la pantalla puede ser desplazada sin una pregunta "Scroll?". Siempre se debe guardar en ella un valor alto, ya que un programa en código máquina se podría parar si se da una respuesta negativa.

ATTR P: Posición 23693 IY + 83 (5C8Dh)

Proporciona los atributos globales para cualquier impresión de pantalla, PLOT, DRAW, etc. Se inicializa con una sentencia de color del BASIC, y puede ser modificada desde dentro de un programa en código máquina; en este caso todos los manejos de colores posteriores cambiarán.

MASK P: Posición 23694 IY + 84 (5C8Eh)

Esta se usa como una máscara entre los elementos transparentes y los de color, tomados de **ATTR P: Posición 23693 IY + 83 (5C8Dh)**. Para cualquier bit puesto a 1, el bit del atributo se tomará desde el atributo de la posición actual en la pantalla; de otro modo se tomará desde ATTR T.

ATTR T: Posición 23695 IY + 85 (5C8Fh)

Esta proporciona los atributos temporales; se actualiza mediante un elemento de color impreso con una instrucción RST 16 (10h).

MASK T: Posición 23696 IY + 86 (5C90h)

Se usa como máscara para discriminar entre los elementos de color transparentes y los de color cogidos de **ATTR T: Posición 23695 IY + 85 (5C8Fh)**. Para cada bit puesto a 1, el elemento se cogerá del atributo de la pantalla de la posición actual; si no, se tomará de ATTR T.

P FLAG: Posición 23697 IY + 87 (5C91h)

Esta es la bandera utilizada para diferenciar entre los parámetros de impresión en cualquier salida a la pantalla realizada por la ROM. Los bits pares son temporales, mientras que los impares son permanentes, ambos referentes a los mismos elementos.

Bit 0/1 está en 1 si se usa OVER 1.

Bit 2/3 está en 1 si se usa INVERSE.

Bit 4/5 está a 1 si se usa INK 9.

Bit 6/7 está a 1 si se usa PAPER 9.

MEMBOT: Posiciones 23698-23727 IY + 88 (5C92h-5CAFh)

Esta es el área usada por el calculador para almacenar los valores que no se pueden guardar convenientemente en la pila del calculador (ver capítulo 8).

No usada: Posiciones 23728/9 IY + 118 (5CBO/1h)

RAMTOP: Posiciones 23730/1 IY + 120 (5CB2/3h)

El uso principal de esta posición, que contiene la dirección del último byte del área BASIC, será, probablemente, para asegurarse de que la pila del calculador tiene espacio suficiente (ver capítulo 8). Se inicializa con una función CLEAR desde el BASIC, que también coloca la base de la pila de la máquina en esta dirección.

P-RAMT: Posiciones 23732/3 IY + 122 (5CB4/5h)

Esta contiene la dirección del último byte de la RAM física.

Variables del sistema 8K

A continuación, están las variables del sistema de la ROM 8K, las cuales sólo existen cuando está conectado el interfaz del Microdrive.

FLAGS3: Posición 23734 IY + 124 (5CB6h)

Bit 0 está a 1 si se está ejecutando una función extendida.

Bit 1 está a 1 si se está ejecutando un CLEAR *.

Bit 2 está a 1 si ha sido alterada la variable del sistema ERR SP por la ROM de 8K.

Bit 3 está puesto a 1 para las rutinas de la red local.

Bit 4 está puesto a 1 para las rutinas LOAD por la ROM de 8K.

Bit 5 está puesto a 1 para las rutinas SA VE por la ROM de 8K.

Bit 6 está puesto a 1 para las rutinas MERGE por la ROM de 8K.

Bit 7 está puesto a 1 para las rutinas VERIFY por la ROM de 8K.

VECTOR: Posiciones 23735/6 IY + 125 (5CB7/8h)

Esta contiene la dirección a la que se salta si la sintaxis falla en las dos ROM, en la de 8K y en la de 16K. Normalmente, señala a 496 (lF0h), la cual causará un error de la ROM de 16K. Se puede cambiar, para señalar a otras rutinas que comprueben más reglas de sintaxis. Esto se explica en el capítulo 7.

SBRT: Posiciones 23737-23746 (5CB9h-5C2h)

No se la puede considerar estrictamente como una variable del sistema, sino una subrutina corta, usada por la ROM 8K para llamar a las rutinas de 16K. Todos los detalles de esta rutina se dan en el capítulo 3, en la rutina RST 16 de la ROM de 8K.

BAUD: Posiciones 23747/8 (5CC3/4h)

Es el valor usado para fijar la velocidad de transmisión (BAUD) para la rutina de entrada/salida RS 232. Se calcula de la forma siguiente:

$$\frac{3500000}{26 * \text{velocidad de transmisión}} - 2 = \text{BAUD}$$

siendo 3500000 la frecuencia de reloj del Spectrum; esto se puede usar para enviar o recibir a velocidades no estándar. El valor que toma por defecto es 12 (0Ch), que da una velocidad aproximada de 9600.

NTSTAT: Posición 23749 (5CC5h)

Este es el número de estación de la red local asignado al Spectrum.

IOBORD: Posición 23750 (5CC6h)

Contiene el color del borde durante las operaciones de entrada/salida, y se carga con el número del color. Normalmente es 0 para negro, pero se puede cambiar.

SER_FL: Posiciones 23751/2 (5CC7/8h)

Se usa durante la entrada RS 232. El primer byte es una bandera colocada a 0 cuando se entra en la rutina ya 1 cuando se recibe un byte. El segundo byte es el recibido cuando se sale de la rutina.

SECTOR: Posiciones 23753/4 (5CC9/Ah)

Se usa por la ROM 8K para contar los sectores del Microdrive.

CHADD_: Posiciones 23755/6 (5CCB/Ch)

Esta es la equivalente en la ROM de 8K de CH_ADD (posiciones 23645/6) de la ROM 16K. Se usa para almacenar la dirección de CH_ADD, mientras que la sintaxis extendida se está comprobando; entonces se sustituye, si es necesario.

NTRESP: Posición 23757 (5CCDh)

NTRESP es el código de respuesta dado a la red local.

Los siguientes 8 bytes forman la cabecera de la red local, que se explica en el capítulo 3.

NTDEST: Posición 23758 (5CCEh)

Aquí está contenido el número de la estación a la que se dirige la salida de la red local.

NTSRCE: Posición 23759 (5CCFh)

Contiene la estación enviada a la red local.

NTNUMB: Posiciones 23760/1 (5CD0/1h)

Esta contiene el número de bloque de la red local que está siendo pasado actualmente.

NTTYPE: Posición 23762 (5CD2h)

Posee la identificación para un bloque de la red local: 0, para un bloque normal, ó 1, para el bloque final.

NTLEN: Posición 23763 (5CD3h)

Esta contiene la longitud del bloque de la red local que se está transmitiendo.

NTDCS: Posición 23764 (5CD4h)

Contiene la suma de comprobación del bloque de datos que va a continuación.

NTHCS: Posición 23765 (5CD5h)

Esta contiene la suma de comprobación de los siete bytes de la cabecera.

Los siguientes ocho bytes forman el primer especificador del fichero.

D_STR1: Posiciones 23766/7 (5CD6/7h)

Para efectuar transacciones al Microdrive, ésta proporciona el número del drive (dos bytes).

Para hacer transacciones de red local, proporciona el número de la estación de destino.

Para transacciones por el RS232, proporciona la velocidad de transmisión. (Para una descripción más amplia del uso de D_STR1, vea el capítulo 3, donde se explican todas las rutinas que la usan.)

S_STR1: Posición 23768 (5CD9h)

Contiene el número de corriente (0-15).

L_STR1: Posición 23769 (5CD9h)

Contiene el tipo de canal en mayúsculas.

N_STR1: Posición 23770/1 (5CDA/Bh)

Almacena la longitud del nombre del fichero.

T_STR1: Posición 23772/3 (5CDC/Dh)

T _STR1 contiene la dirección de inicio del nombre del fichero. Los siguientes 8 bytes los usan por las funciones LOAD y MOVE.

D_STR2: Posiciones 23774/5 (5CDE/Fh) a T_STR2: Posición 23780/1 (5CE4/5h)

Estos 8 bytes son los mismos que forman el primer especificador del fichero.

Los bytes siguientes son equivalentes directos a los bytes de la cabecera de las rutinas de la ROM 16K, pero son usados por la ROM 8K (una exposición de cómo usarlos se da en el capítulo 3).

HD_00: Posición 23782 (5CE6h)

Este posee el tipo de fichero donde: 0 es programa; 1, matriz numérica; 2, matriz alfanumérica, y 3, código (code).

HD_0B: Posiciones 23783/4 (5CE7/8h)

La longitud de los datos se encuentra en estas direcciones.

HD_0D: Posiciones 23785/6 (5CE9/Ah)

Aquí se encuentra el inicio de los datos.

HD_0F: Posiciones 23787/8 (5CEB/Ch)

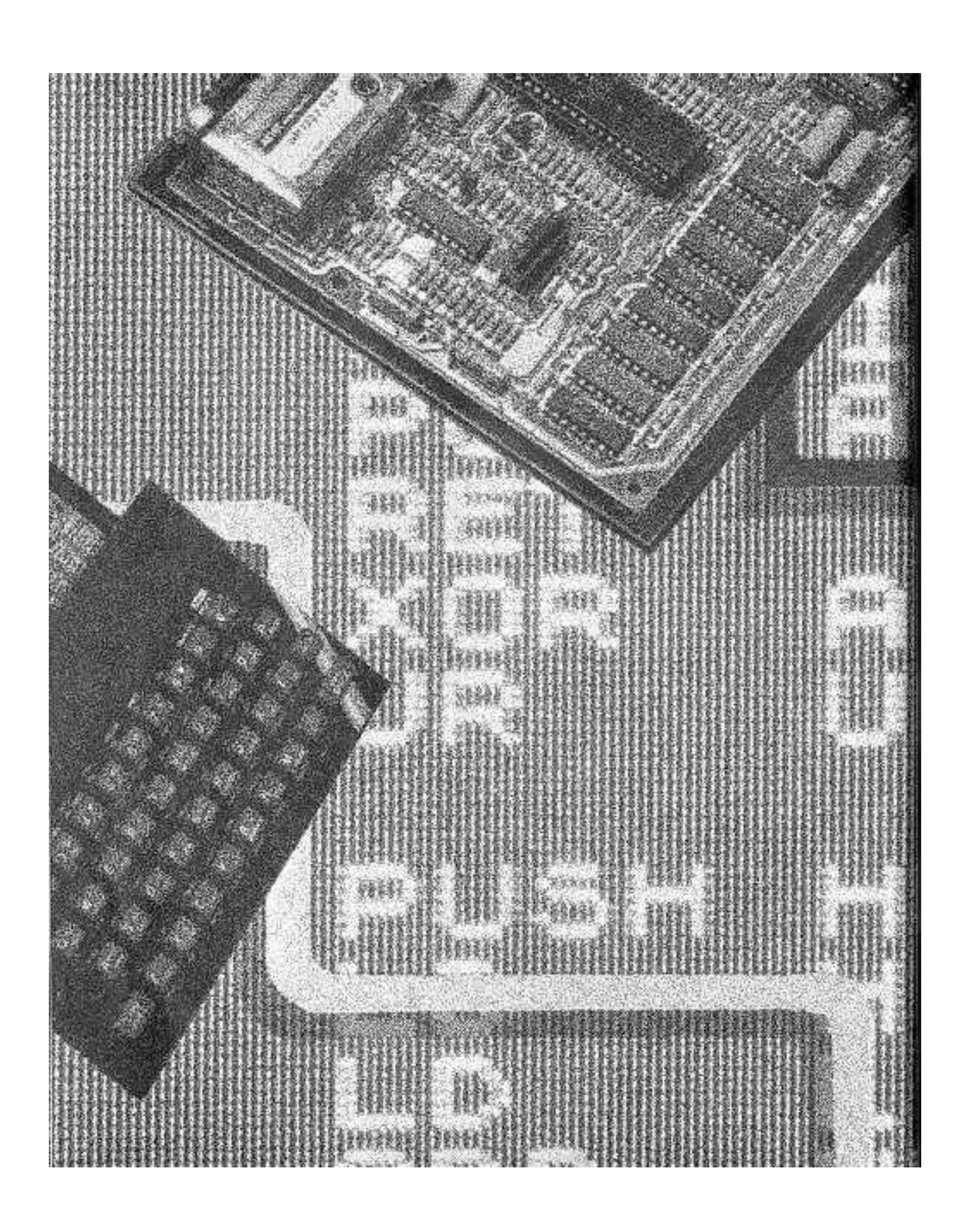
Posee el nombre de la matriz o la longitud del programa.

HD_11: Posiciones 23789/90 (5CED/Eh)

En ésta se almacena el número de línea de inicio automático. Se usa también por el código de enlace 32h (ver capítulo 3).

COPIES: Posición 23791 (5CEFh)

Indica el número de copias hechas por una instrucción SAVE.


PUSH

5. Puertos y canales de entrada y salida

El Spectrum estándar tiene todas las líneas de direcciones y de datos en el conector de la parte posterior, y el BASIC permite la comunicación con el mundo exterior por medio de los comandos IN y OUT. Las instrucciones IN A, (C) y OUT A, (C) de la CPU Z80 permiten que se pueda usar todo el bus de direcciones para determinar qué periférico está siendo controlado.

El puerto 254 (FEh) se usa como salida para fijar el color del borde (BORDER), para la salida de la cinta y para hacer sonar el altavoz interno. También maneja el teclado y la entrada de la cinta. Un breve resumen de esto se da en el capítulo dedicado a la ROM de 16K, y lo explicaré completamente más adelante en este mismo capítulo.

El puerto 251 (FBh) maneja la impresora ZX y es usado para la entrada y salida.

El puerto 247 (F7h) pasa datos para la red local y para la entrada/salida de las comunicaciones RS232.

El puerto 239 (EFh) controla el Microdrive y el protocolo en la entrada/salida RS232 del Interfaz 1.

El puerto 231 (E7h) maneja los datos del Microdrive en lectura y escritura.

Este último puerto es la razón por la que algunos periféricos son incompatibles con el interfaz del Microdrive. El manual del Spectrum no menciona que vaya a ser usado. Ahora se explicarán los detalles de los puertos más utilizados.

Puerto 254 (FEh) 11111110 BIN

Por este puerto se lee el teclado en los bits del 0 al 4. Cada línea de teclas se divide en dos secciones de cinco teclas, y en cada sección la tecla exterior se lee en el bit 0 y la tecla más cercana al centro en el bit 4. Cada bit es devuelto en estado alto (1), a menos que una tecla haya sido pulsada; en tal caso, el bit respectivo está en estado bajo (0).

Las líneas de dirección se usan para determinar qué sección es leída por una instrucción IN. Para colocar a 0 el bit de cada línea se pone como sigue:

A0	CAPS SHIFT - V	254 (FEH)	11111110
Al	A - G	253 (FDH)	11111101
A2	Q - T	251 (FBH)	11111011
A3	1 - 5	247 (F7H)	11110111
A4	0 - 6	238 (EFH)	11101111
A5	P - Y	223 (DFH)	11011111
A6	ENTER - H	191 (BFH)	10111111
A7	BREAK/SPACE - B	127 (7FH)	01111111

Desafortunadamente, los bits del 5 al 7 de los datos leídos están en un estado no predecible y, de hecho, la lectura del teclado ha sido cambiada en la "ISSUE 3" del Spectrum, creando algunos problemas en los programas que se emplean para examinar el teclado, pero no excluyen estos bits. La rutina para explorar el teclado del Spectrum los ignora. Este último factor abre la posibilidad de que sean usados para otros fines, por ejemplo, para definir teclas de funciones como las existentes en el BBC y otros ordenadores, manejadas por una rutina basada en interrupciones.

Es posible leer más de una línea del teclado a la vez colocando los bits a 0 antes de leer en las líneas de datos, pero esto no permite la discriminación entre las diferentes líneas de teclas.

Nota: Es necesario prestar atención cuando se estén leyendo varias teclas pulsadas al mismo tiempo. Sinclair no ha puesto ninguna protección contra la realimentación entre las líneas del teclado. Esto significa que si, por ejemplo, las teclas 'A', 'S' y 'W' son pulsadas al mismo tiempo, una exploración de la línea de la Q a la T mostrará que Q ha sido pulsada, aunque no sea así. Esto sucede porque mientras se pulsan dos teclas en diferentes líneas de dirección, pero con el mismo bit de datos, las líneas se conectan entre sí; por tanto, cualquier otra tecla pulsada en otra línea colocará a 0 el bit respectivo en ambas líneas durante el tiempo que las otras teclas se mantengan pulsadas.

El bit 6 es la entrada del conector del cassette, y tiene una tendencia a colocarse a 0, aunque puede ser colocado a 1 por una instrucción OUT. Es necesario tener cuidado, ya que cualquier salida a los bits 0-2 pondrá el color del borde. El método más seguro de colocar todos los bits a 1 es la salida 248 (F8h), pero (1) esto sólo será temporal, y (2) no será nunca necesario si se hace una decodificación correcta.

En salida, los bits 0, 1 y 2 controlan el color del borde: el bit 0 controla el azul; el bit 1, el rojo, y el bit 2, el verde; todos los demás colores se forman de la mezcla de estos tres.

Color	Número	Binario
NEGRO	0	00000000
AZUL	1	00000001
ROJO	2	00000010
MAGENTA	3	00000011
VERDE	4	00000100
AZULADO	5	00000101
AMARILLO	6	00000110
BLANCO	7	00000111

El bit 3 controla el conector del micrófono. Recuerde que para conseguir algo más que un click necesita ser encendido y apagado repetidamente para conseguir un tono. El MIC impreso en la parte de atrás del Spectrum está ligeramente equivocado, dado que proporciona salida a la entrada de MIC de un cassette, y enchufar un micrófono resultará una pérdida de tiempo.

El bit 4 controla el altavoz interior del Spectrum, y son válidos los mismos comentarios anteriores.

Un programa que demuestra cómo usar el altavoz interno y la entrada EAR para leer palabras o música, almacenándolas en la memoria y permitiendo la repetición de la secuencia a través del altavoz, se da al final de este capítulo.

Puerto 251 (FBh) 11111011 BIN

Bit 0 ENTRADA es la línea de ocupada de la impresora ZX; puesta a 0 está ocupada.

Bit 1 SALIDA alta (1) retarda el motor; bajado a 0, la velocidad sube de nuevo.
Bit 2 SALIDA baja (0) pone en marcha el motor.

Bit 6 ENTRADA alta (1) si la impresora no está conectada.

Bit 7 SALIDA alta (1) hace la impresión; un bit cada vez.

Puerto 247 (F7h) 11110111 BIN

Bit 0 SALIDA datos en serie para red local y RS232. ENTRADA de datos en serie desde la red local.

Bit 7 ENTRADA de datos en serie desde RS232.

Puerto 239 (EFh) 11101111 BIN

El saliente de protección contra escritura en un cartucho de Microdrive puede comprobarse con IN A, (239) AND 1; la bandera 0 estará puesta a 1 si el cartucho está protegido.

La presencia de un drive puede ser buscada leyendo el bit 2 del puerto 239, después de seleccionar la unidad; estará a 0 si existe.

La línea RS232 DTR está en el bit 3 del puerto 239, y la línea CTS está en el bit 4 de este mismo puerto.

Es muy improbable que se necesite usar los puertos 247, 239 y 231 desde otro lugar que no sea dentro de la ROM de 8K, con la excepción de la comprobación de existencia de un mecanismo auxiliar.

Los puertos restantes están disponibles para ser usados por otros mecanismos auxiliares. Recuerde que si una impresora y otro interfaz están conectados usarán un puerto o puertos para la transferencia de información. Dos de los interfaces de impresora en paralelo más comunes son el MOREX (que también tiene un interface bidireccional RS232, y yo lo recomiendo) y el Kempston, que utilizan los siguientes puertos:

Puertos Morex 251 (FBh) 11111011 BIN y 127 (7Fh) 01111111 BIN

Puerto 251 (FBh) 11111011 BIN

Bits 0- 7 SALIDA de datos Centronics.

Bit 0 ENTRADA Centronics ocupada.

Bit 1 ENTRADA RS232 DTR.

Bit 7 ENTRADA de datos RS232 RX.

Puerto 127 (7Fh) 01111111 BIN

Bit 0 SALIDA señal Centronics de dato enviado.

Bit 1 SALIDA RS232 de datos TX.

Bit 2 SALIDA RS232 CTS.

Puerto Kemspton 58047 (E2BFh), 57535 (EOBFH) y 58303 (E3BF)

El interfaz de Kempston puede ser usado sólo con las instrucciones IN (C) y OUT (C), dado que comprueba los 16 bits del bus de direcciones. Las rutinas para enviar un solo carácter a cada uno

de estos interfaces de salida Centronics se pueden encontrar en el apéndice G, **Subrutinas útiles**.

Puerto 58047 (E2BFh)

Bit 0 ENTRADA indica que la línea está ocupada.

Puerto 57535 (EOBFh)

Bits 0- 7 SALIDAS de datos por Centronics.

Puerto 58303 (E3BFh)

Bits 0-3 SALIDAS señal Centronics de dato enviado.

Corrientes estándar

Corriente K- 3/253 (FDh) es un duplicado de las corrientes 0 y 1. Corriente S - 2/254 (FEh) es un duplicado de la corriente 2.

Corriente R- 1/255 (FFh) se usa para escribir en un área de trabajo, y pone el código contenido en el registro A dentro de la dirección contenida en la variable del sistema K CUR 23643/4 (5C5B/Ch), e incrementa la dirección en K CUR. Esto no es tan útil como parece en principio, dado que la rutina para reservar espacio es llamada primero, y ésta desplaza toda la memoria sobre la dirección en K CUR hasta RAMTOP, inutilizando la corriente para poner alguna cosa en algún lugar sobre RAMTOP, o en algún sitio que no debe ser movido. La rutina se inicia en 3969(F81h), y la corriente puede ser utilizada sólo para salida; cualquier intento para usarla como entrada causará un error. Las corrientes 0 y 1 (K) son vinculadas, normalmente, a la parte inferior de la pantalla y al teclado. También permiten realizar entrada por sus canales. Corriente 2 (S) es sólo para salida; normalmente, a pantalla.

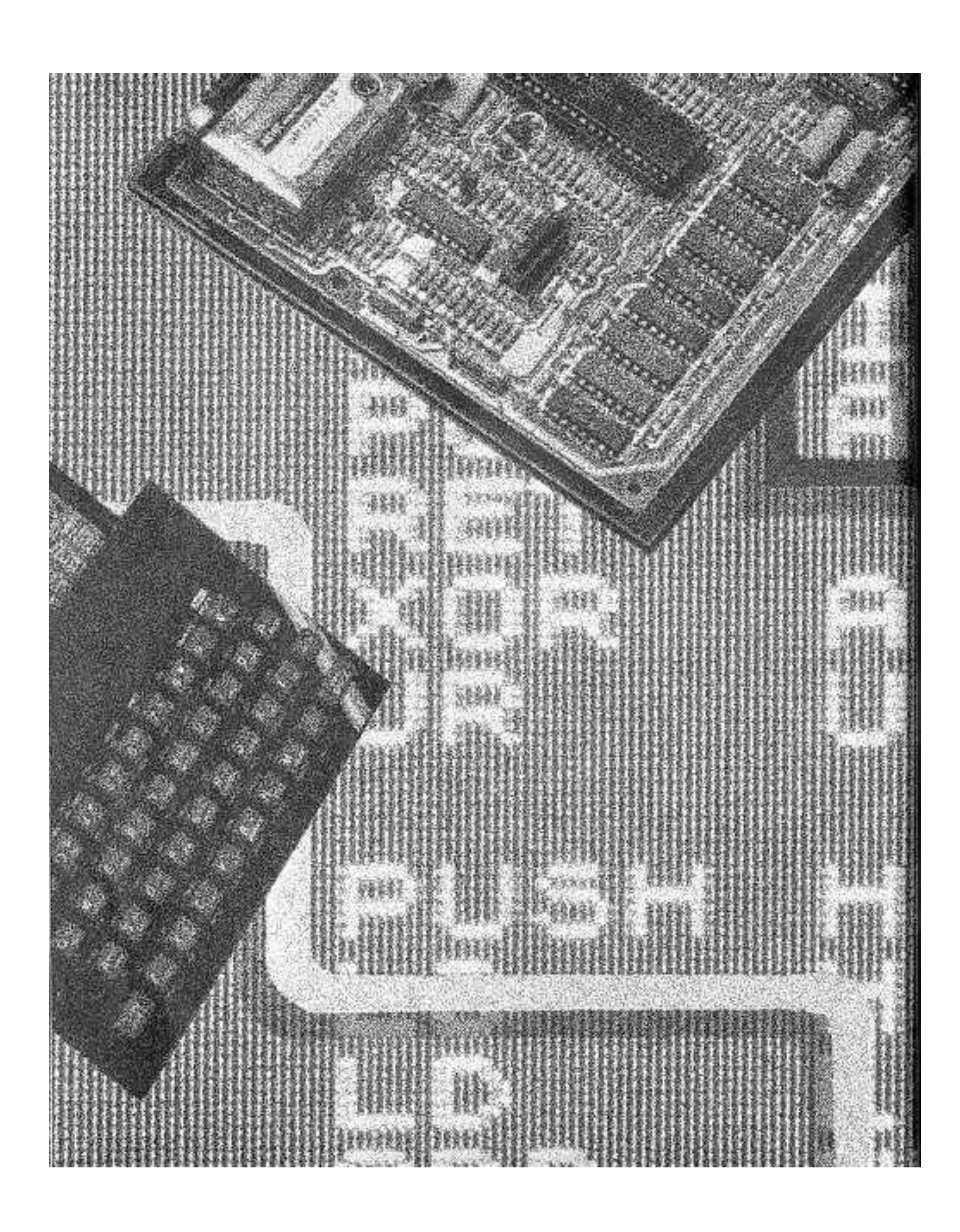
Corriente P 3 es sólo para salida; normalmente, vinculada a la impresora. Hay 19 corrientes disponibles, y cada una tiene que ser vinculada a un canal; cada corriente tiene dos bytes en el área de las corrientes de las variables del sistema, comenzando en 23569 (5C10h) para la corriente - 3, conteniendo el desplazamiento del canal al que está vinculada desde la base del área de los canales. Recuerde que la primera corriente es - 3, por lo que para encontrar la dirección del canal para la corriente 0 tendrá que mirar en 23574/5 (5C16/7h).

Un canal está compuesto por un mínimo de cinco bytes; los dos primeros poseen la dirección de la rutina de salida; los dos siguientes contienen la dirección de la rutina de entrada, y el último byte contiene las letras mayúsculas del código del canal (S, K, P, etc.). Los canales asociados con el interfaz del Microdrive son sustancialmente más grandes que el mínimo, y su formato se muestra en el manual proporcionado con el interfaz.

Rutina de grabación y reproducción

```
                                    ; este corto programa permite
                                    ; oír una voz por la entrada
                                    ; EAR y meterlo en la memoria.
                                    ; El texto puede ser reproducido
                                    ; luego por el altavoz del Spectrum.
                                    ; es un programa muy sencillo
                                    ; pero funciona.
                                    ;
C350        LIMITE   EQU 50000      ; límite inferior de la memoria
0005        TIEMPO   EQU 5          ; retardo entre cada bit.
```

```
FDE6            110  PRIMER  EQU 64998        ; El primer byte que se va a usar.
FDE8            120          ORG 65000
FDE9  21E6FD    130  INICIO  LD HL, PRIMERO   ; calcula la memoria
FDEB  1150C3    140          LD DE, LIMITE    ; disponible para su uso.
FDEE  E5        150          PUSH HL
FDEF  A7        160          AND A
FDF0  ED52      170          SBC HL, DE
FDF2  E5        180          PUSH HL
FDF3  C1        190          POP BC
FDF4  E1        200          POP HL
FDF5  F3        210          DI               ; desconecta las interrupciones para
                220                           ; tener una temporización correcta
FDF6  3E7F      230  ESCUCH  LD A, #7F        ; espera un sonido
FDF8  DBFE      240          IN A, (#FE)
FDFA  CB77      250          BIT 6, A
FDFC  28F8      260          JR Z, ESCUCHA
FDFE  C5        270          PUSH BC
FDFF  0608      280  TOMABY  LD B, 8          ; lee 8 bits
FE01  2B        290          DEC HL
FE02  DBFE      300  OYE     IN A, (#FE)      ; un bit cada vez
FE04  17        310          RLA
FE05  17        320          RLA
FE06  CB16      330          RL (HL)          ; y la transfiere a la memoria
FE08  100A      340          DJNZ PAUSA       ; pausa y coge el siguiente bit
FE0A  D1        350          POP DE           ; pero si se han leído 8 mira
FE0B  1B        360          DEC DE           ; si hay sitio y decrementa el contador
FE0C  7A        370          LD A, D
FE0D  B3        380          OR E
FE0E  D5        390          PUSH DE
FE0F  20EE      400          JR NZ, TOMABY    ; si hay sitio vuelve a coger
FE11  C1        410          POP BC           ; otro byte, si no, acaba.
FE12  FB        420          El
FE13  C9        430          RET
FE14  0E05      440  PAUSA   LD C, TIEMPO
FE16  0D        450  ESPERA  DEC C
FE17  20FD      460          JR NZ, ESPERA
FE19  18E7      470          JR OYE
FE1B  21E6FD    480  HABLA   LD HL, PRIMERO   ; es lo mismo que para escuchar
FE1E  1150C3    490          LD DE, LIMITE
FE21  E5        500          PUSH HL
FE22  A7        510          AND A
FE23  ED52      520          SBC HL, DE
FE25  E5        530          PUSH HL
FE26  C1        540          POP BC
FE27  E1        550          POP HL
FE28  C5        560          PUSH BC
FE29  F3        570          DI
FE2A  0608      580  SACABY  LD B, 8          ; saca 8 bits
FE2C  2B        590          DEC HL
FE2D  7E        600          LD A, (HL)       ; notar que se sacan todos los bits y
FE2E  0F        610          RRCA             ; esto cambiara el color de BORDER
FE2F  0F        620          RRCA
FE30  0F        630          RRCA
FE31  D3FE      640  SACABI  OUT (#FE), A     ; un bit cada vez, como al leer
FE33  07        650          RLCA
FE34  0E05      660          LD C, TIEMPO
FE36  0D        670  RETARD  DEC C
FE37  20FD      680          JR NZ, RETARDO
FE39  10F6      690          DJNZ SACABI
FE3B  C1        700          POP BC
FE3C  08        710          DEC BC
FE3D  79        720          LD A, C
FE3E  80        730          OR B
FE3F  C5        740          PUSH BC
FE40  20E8      750          JR NZ, SACABY
FE42  C1        760          POP BC
FE43  FB        770          EI
FE44  C9        780          RET
```


PUSH

6. Usando las interrupciones

La secuencia de encendido en el Spectrum, que borra la memoria y coloca las variables del sistema, también inicializa el registro de interrupciones para contener 63 (3Fh) y poner el modo de interrupción a 1 (IM1). La modificación del registro I parece innecesaria, dado que el modo IM1 no lo usa -porque en este modo cualquier interrupción hace un RST 56 (38h)-, pero la innovadora forma en la que la ULA maneja la pantalla en el Spectrum hace importantes los bits 6 y 7 del registro I.

En cualquier ciclo de instrucción de máquina, el microprocesador Z80 ejecuta una operación para refrescar la memoria durante la cual los contenidos del registro I se sacan en los ocho bits altos del bus de direcciones y la línea de petición de memoria se activa. La ULA genera una interrupción cada vez que desea actualizar la pantalla. Esto hace que la CPU ejecute la rutina de servicio de interrupción, asumiendo que las interrupciones estén habilitadas. Lo normal es que sea la exploración del teclado y la actualización del contador FRAMES, pero si la ROM del Interfaz está conectada y activa, sucede que se habilitan las interrupciones y se produce un retorno inmediato sin explorar el teclado o hacer ninguna otra cosa.

Cuando la rutina de interrupciones se completa, la CPU retorna a lo que estaba haciendo previamente. Si esto incluye una instrucción de lectura o escritura en una memoria entre 16384 (4000h) y 32767 (7FFFh), que la ULA comprueba mirando las dos líneas superiores del bus de direcciones, y la línea MREQ, la ULA para el reloj en la CPU mientras completa la actualización de la pantalla.

Si el registro I tiene el bit superior puesto a 0 y el bit 6 a 1, la ULA se confunde, debido al refresco de la memoria dinámica durante T3 y T4 de un ciclo M1 (de coger instrucción de la memoria). Esto activa la línea MREQ, y hace que el registro I salga en los ocho bits superiores del bus de direcciones. La ULA entonces cree que la CPU está haciendo una operación de lectura o escritura en este área de la RAM, aun cuando haya intentado pararla, y la ULA omite su propia lectura para actualizar la pantalla, produciendo una interferencia en la pantalla. El registro I no puede, por tanto, poseer un valor que tenga los dos bits superiores dispuestos de este modo; en otras palabras, cualquier número desde el 64 al 127 (40h a 7Fh) inclusive, si se quieren evitar interferencias en la pantalla.

Modificando IM2 puede usar las interrupciones para sus propios fines, mientras que usted haga una RST 56 (38h) al final de su propia rutina de servicio de interrupción, la cual habilitará las interrupciones antes de retornar al programa de llamada -si usted quiere explorar el teclado y actualizar el contador y terminar con una instrucción RETI.

Si usted no ha usado la RST 56 (38h) dentro de la rutina de interrupción, debe ejecutar una instrucción El antes de la RETI, si quiere habilitar las instrucciones para llamar de nuevo a la rutina de servicio de interrupción. Recuerde que tendrá que volver a colocar el modo IM1 y permitir las interrupciones antes de regresar al BASIC, a menos que usted esté usando una RST 56 (38h) dentro de la rutina de interrupción.

El modo IM2 es algo enrevesado. Al recibir una interrupción, que se produce 50 veces por segundo (60 en Estados Unidos), la CPU salva la dirección de la próxima instrucción del programa que está ejecutando en la pila de la máquina, e imposibilita cualquier otra interrupción. Entonces mira a la posición señalada por el bus de datos + (256 * el registro I) y salta a la dirección que está contenida en esa posición + (256 * el contenido de la posición siguiente). Se considera normalmente como una mala práctica el tener el bit 0 en el bus de datos en estado alto usándolo como puntero en IM2, dado que el vector debería comenzar siempre en una dirección par, pero desafortunadamente con el Spectrum no hay elección.

Ejemplo: El registro I contiene 10 (0Ah), y el bus de datos contendrá 255 (FFh). 10 * 256 = 2560 y 2560 + 255 = 2815; por tanto, la dirección de salto será tomada de los contenidos de la dirección 2815 + (256 * los contenidos de la dirección 2816). La dirección 2815 contiene 34 y la dirección 2816 contiene 1281. Esto lo puede comprobar por sí mismo haciendo un PEEK en su Spectrum, ya que éstas están en la ROM; por tanto, la dirección de salto será 34 + (256 * 128), que es 32802.

De igual forma, si el registro I contiene 6:

6 * 256 = 1536 y 1536 + 255 = 1791.
1791 contiene 221 y 1792 contiene 113.
...221 + (113 * 256) = 29149; por tanto, el salto será a 29149.

Alternativamente, si usted tiene un Spectrum 48K, y el registro I poseyera 200: 200 * 256 = 51200 y 51200 + 255 = 51455; de modo que el salto sería a la dirección que usted pusiera dentro de esta posición y de la siguiente, en el modo normal del Z80 con el byte inferior primero.
Esto se puede representar imaginando la interrupción como una instrucción invisible en el programa que está siendo ejecutado. En el momento de la interrupción, la instrucción invisible es ejecutada como si fuera una DI seguida por una instrucción de llamada (CALL), a la dirección inmediatamente anterior a la señalada por el registro I y el bus de datos; al ser llamada, la dirección está en los dos bytes siguientes en el orden del primer byte inferior del Z80 estándar. La instrucción, al ser invisible, no puede colocar su propia dirección de retorno en la pila de la máquina; por consiguiente, la dirección posterior a la de la última instrucción ejecutada en su programa, se guarda en la pila y es a esta dirección a la que volverá el control después a la instrucción RETI, al final de la rutina de interrupción.

La instrucción RETI debe ser precedida por una instrucción El. La razón de que la DI sea incorporada en la llamada ejecutada por la interrupción es para asegurarse de que, si el tiempo de ejecución de la rutina de servicio de interrupción es más largo que la dilación entre dos interrupciones, el programa no se quede encerrado en un bucle.

Es muy fácil escribir un programa que cambie la dirección de salto en una interrupción cargando los bytes del vector (las dos direcciones miradas para determinar dónde se hace el salto) con la dirección deseada dentro del programa.

Nota: Siempre que las rutinas de interrupción sean usadas es de vital importancia que todos los registros utilizados por ellas se guarden al entrar y se recuperen antes de volver al programa principal. No intente pasar datos desde o hacia la rutina de interrupción en los registros.

Debido a la limitación de los valores que puede contener el registro I, sólo hay un número limitado de direcciones a las que se puede saltar en el Spectrum de 16K, y ésas las fijan los contenidos de la ROM. El interfaz del Microdrive crea un nuevo problema cuando se usan las interrupciones basadas en ROM, ya que cambia el vector siempre que está conectado en el apéndice F se da una lista de los vectores para la "ISSUE 2" del Spectrum y la "ISSUE 1 " del interfaz del Microdrive; pero si no está seguro de su versión, o sí tiene una versión diferente, esto debe ser comprobado. En los programas comerciales es peligroso usar vectores de la ROM, ya que cualquier cambio o adiciones futuras podrían inutilizar su programa.

El uso típico de las rutinas de interrupción es controlar sprites y sonido continuo dentro de un programa. Sabiendo cada cuánto tiempo se genera una interrupción, es fácil calcular la velocidad del movimiento de un sprite e, independientemente de cualquier otra operación dentro del programa, la velocidad permanecerá constante.

El uso de las rutinas de la ROM dentro de una rutina de interrupción se complica debido a la posibilidad de que la ROM del interfaz esté conectada cuando se produce la interrupción y esto debe ser permitido cuando se escribe la rutina. Por ejemplo, si una rutina SPRITE usa una llamada de la ROM de 16K para dibujar el sprite en la pantalla por medio de la llamada de PLOT en 8933 (22E5h) cuando la ROM del interfaz está activa, la llamada será a la dirección 8933 en la ROM del interfaz. Esta es una dirección que no existe, y causará, inevitablemente, el fallo del programa.

Una forma de solventar el problema es incorporar dentro de la rutina de interrupción un chequeo para ver que la ROM está presente. El método más fácil de chequear es mirar en una dirección de la ROM que contenga un valor diferente en cada una. Yo suelo usar la dirección 20 (14h), la cual contiene 213 (D5h) en la ROM del interfaz y 255 (FFh) en la ROM de 16K. Entonces se puede adoptar la acción apropiada para cada ROM.

Si la llamada de la ROM es para la ROM de 16K, se puede llamar directamente cuando ésta esté presente, y mediante la instrucción RST 16 cuando la ROM de 8K esté conectada. La ROM de 8K se puede utilizar con el código de enlace 32h (los detalles se dan en el capítulo 3). En el apéndice G se da un sencillo programa de SPRITES que mueve un grupo de cuatro puntos, haciéndolos botar en los cuatro bordes de la pantalla, emitiendo un sonido y cambiando el color del borde, mostrando métodos de resolver el problema.

Para una comprensión total de las interrupciones, si va a hacer algún uso de ellas, le sugiero que introduzca el programa usando su ensamblador. El listado está dado directamente desde nuestro ensamblador, para asegurarse de que no hay errores; es algo extraño en el hecho de que los números hexadecimales están precedidos por un #, y los números binarios por un % en el código fuente. La rutina se puede relocalizar si se calcula un nuevo vector y el registro I se cambia en consecuencia. Una vez que usted haya entrado y ensamblado el programa, antes de intentar ejecutarlo, siga leyendo.

El primer problema con el que usted se puede encontrar es, si tiene un Spectrum de 16K, que el vector tendrá que estar basado en ROM. Desgraciadamente, cuando estoy escribiendo esto, no he encontrado el modo de usar un Spectrum de 16K con el interfaz del Microdrive.

Antes de hacer cualquier cosa, salve el código fuente y el código objeto en cinta o Microdrive y, si tiene un Microdrive, saque el cartucho. Para iniciar el SPRITE, debe llamar a la rutina etiquetada SETUP del listado. Si el vector no ha sido cambiado, RANDOMIZE USR 51457. Ahora debería ver un punto negro moviéndose en la pantalla; si no lo ve, debe comprobar su código de nuevo. La presencia del SPRITE no debería afectar a ninguna otra cosa que esté haciendo el ordenador, de modo que puede ser introducido un programa y ejecutado normalmente mientras que no pase los límites de la memoria usada por la rutina de la interrupción.

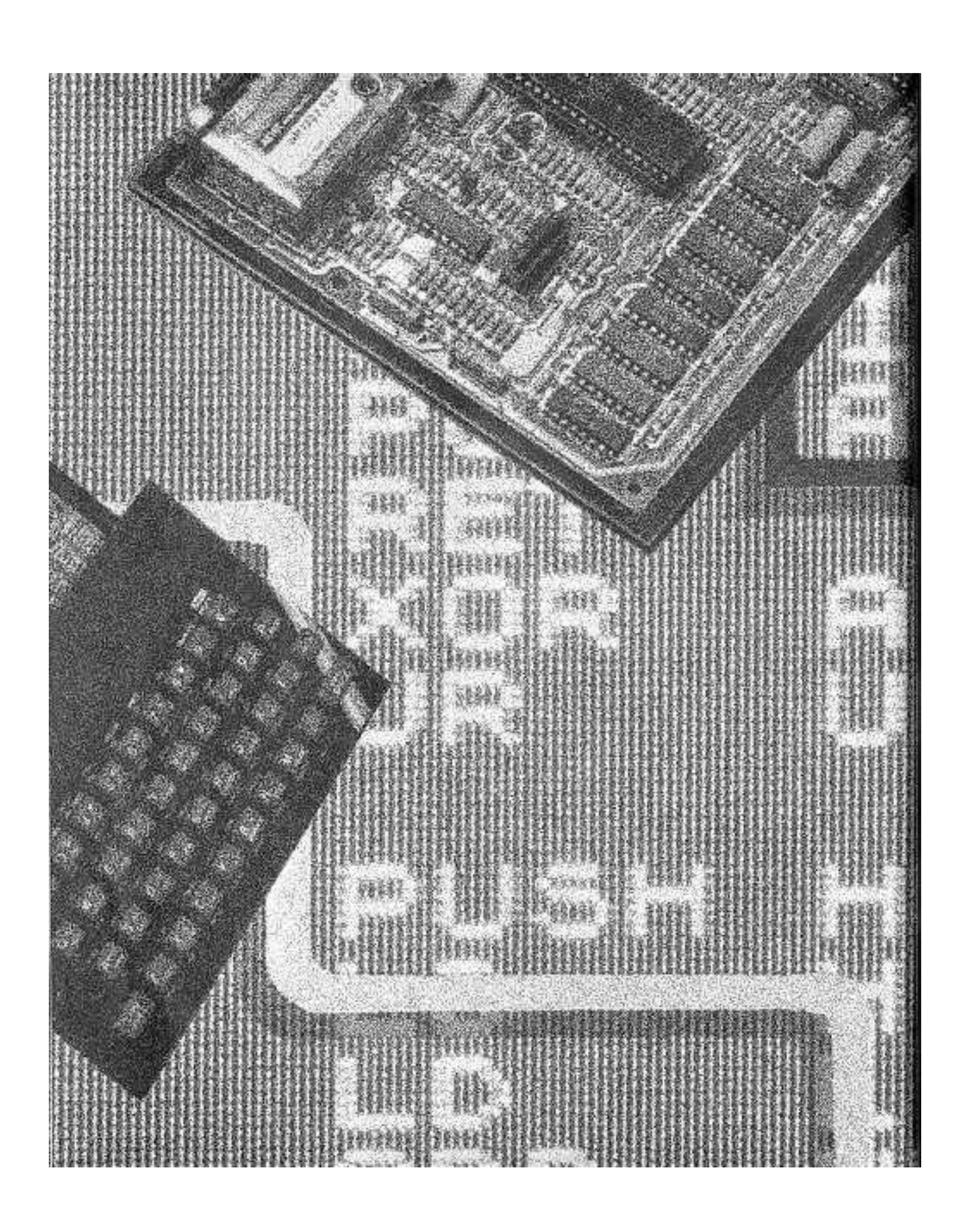
Algunas de las instrucciones del BASIC que actúan sobre las interrupciones pueden demostrarse ahora. Introduzca la línea de BASIC:

```
10 BEEP 5,60: FOR N = 1 TO 100: NEXT N: GOTO 10
```

Cuando está se ejecute, verá que el SPRITE se para, mientras que el BEEP se está ejecutando. Otros comandos que inhiben las interrupciones son los que salvan y cargan.

El sprite se desplazará a la velocidad de 50 pixels por segundo en Europa (60 pixels por segundo en Estados Unidos); por tanto, en España tardará 3,5 segundos para desplazarse desde la parte inferior de la pantalla a la parte superior.

Para más información sobre el diseño de sprites se recomienda consultar el libro Sprites y gráficos en lenguaje máquina para ZX Spectrum (Durst, Anaya Multimedia, 1985).


PUSH

7. Extendiendo el BASIC con el Interfaz 1

Con la adición del Interfaz 1, cualquier instrucción de BASIC que falle la comprobación de sintaxis del Spectrum es, normalmente, dirigida a las rutinas de manejo de error mediante la dirección contenida en la nueva variable del sistema VECTOR. Esta puede modificar se para señalar a una dirección de la RAM, lo cual permite escribir un programa que compruebe la sintaxis nuevamente y ejecute las instrucciones de que esté programado para tratar. Antes de que se puedan usar estas facilidades es necesario comprender el método por el que se comprueban las líneas, de modo que la rutina extra pueda hacer las mismas comprobaciones.

Tan pronto como se pulse la tecla ENTER después de introducir una línea del BASIC dentro de la memoria intermedia de la entrada, el intérprete de la ROM entra en acción. Este ejecuta la línea, pero no llega a ejecutar los comandos, porque la bandera de sintaxis está bajada (bit 7 en FLAGS). Si ocurre cualquier error, es señalado con una '?' y debe corregirse el error antes de que la línea pueda ser insertada en un programa.

El mismo proceso se lleva a cabo en el momento de la ejecución, pero dado que la bandera de sintaxis está levantada, el comando es ejecutado.

Recuerde que siempre que una rutina en la RAM se usa como parte del intérprete BASIC, la ROM de 8K estará paginada.

Hay rutinas en ambas memorias ROM que comprueben la bandera de sintaxis y vuelven a la rutina que hace la llamada sólo cuando se está ejecutando. Estas son más útiles cuando se añaden comandos, como se demuestra en las rutinas al final de este capítulo. El criterio básico para comprobar la sintaxis se da a continuación, y las rutinas de la ROM han sido usadas hasta donde ha sido posible.

La dirección del carácter en la que falla la sintaxis, en lo referente a la ROM, estará presente en la variable del sistema CH_ADD al llegar al comprobador ampliado de la sintaxis en la RAM. Por esta razón es importante asegurarse de que el primer carácter del comando agregado al BASIC rompe la sintaxis normal. Si éste no fuera el caso, el intérprete iniciaría la ejecución al mismo tiempo, y sería casi imposible obtener de nuevo el control. Por esta razón, es más simple usar un carácter no alfabético al inicio del comando adicional, que hace que el sistema se salga del modo normal de órdenes (K), asegurando el error de la sintaxis. Los símbolos, *, y '!' son ideales para esto.

Lo primero que se debe comprobar son los caracteres del comando. Esto se hace tomando cada uno de los caracteres en sucesión y asegurándose de que es el esperado. La rutina de la ROM de 16K en 24 (18h) guardará el carácter que se está tratando en el registro A, y la rutina en 32 (20h) cogerá el carácter siguiente e incrementará CH_ADD. Puede probarse entonces cada carácter y, si no es correcto, se llama a la rutina de manejo de errores saltando al VECTOR original, que es 496 (lF0h).

Entonces:

1. Si hubiera una expresión numérica a continuación, se usaría la rutina de 16K en 7298 (lC82h). Esta evalúa la siguiente expresión como numérica; si no es de este tipo, causará un error. En el momento de ejecutar pondrá también el valor dentro de la pila del calculador.

2. Si hubiera dos expresiones numéricas juntas separadas por una coma, se usaría la rutina de 16K en 7290 (lC7Ah). Esta actúa como la anterior, pero acumula dos números cuando se está ejecutando.

3. Si a continuación debe haber una expresión alfanumérica (entre comillas o una variable terminada en $), entonces se puede usar la rutina de 16K en 7308 (lC8Ch). Esta funciona como las anteriores, pero generando un error si la expresión no es alfanumérica y durante la ejecución almacena los parámetros como se describe al final del capítulo 8.

Nota: En las tres rutinas anteriores, el primer carácter de la expresión a evaluar debe estar en el registro A, y CHADD debe contener su dirección antes de llamar a la rutina. Después de la evaluación, el registro A contendrá el primer carácter después de la expresión, y CHADD su dirección. Se permiten los operadores matemáticos, así como el uso de variables en las expresiones evaluadas, y esto se puede usar como una ventaja para encontrar datos de variables. Por ejemplo, la expresión X$(1,4 TO 9) o A * (B + (C / (D + E))) podrían ser válidas, y el resultado sería apilado en el momento de la ejecución.

Cuando la evaluación se completa, un CALL 1463 (5B7h) comprobará que el carácter actual es el indicador de final (dos puntos o el código de un retorno de carro: 13 (0Dh)) y vuelve al intérprete durante la comprobación de sintaxis o a su rutina para ejecutar, si el programa está funcionando; si no es ninguno de estos caracteres, genera un error.

En el momento de ejecutar, su rutina tendrá que realizar la orden, cogiendo información de la pila del calculador, si es necesario, y volviendo al intérprete mediante una instrucción IP 1473 (5C 1H), con CHADD señalando al carácter siguiente que se tiene que interpretar.

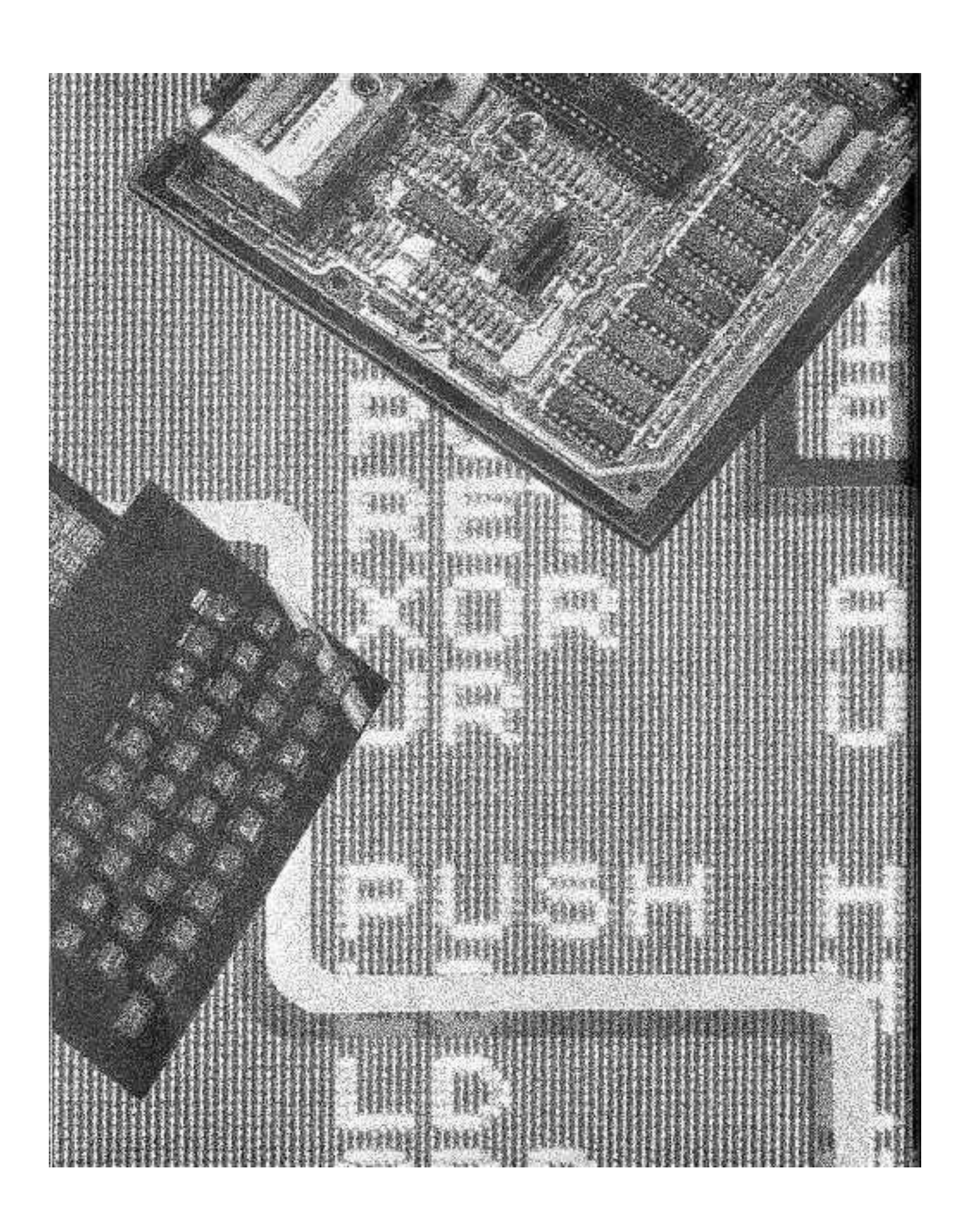
Para demostrar la adición de los comandos, el siguiente programa añade !CALLnn y !FRE !CALLnn llamará a una rutina de código máquina en la dirección indicada a continuación de !CALL !FRE devolverá un número indicando la memoria libre.

```
A91C   D7              CALL    RST     16
A91D   1800                    DEFW    24          ; coge un carácter
A91F   FE21                    CP "!"              ; ¿es una admiración?
A921   2032                    JR NZ, ERROR        ; si no, produce error
A923   CD87A9                  CALL SIGUIE         ; coge el siguiente Carácter
A926   FE43                    CP "C"              ; es una C
A928   2017                    JR NZ, FREE         ; si no, salta y mira
                                                   ; si se trata de un !FREE
A92A   CD87A9                  CALL SIGUIE
A92D   FE41                    CP "A"              ; se comprueba cada carácter
A92F   2024                    JR NZ, ERROR        ; se produce un error
A931   CD87A9                  CALL SIGUIE         ; si no coincide alguno
A934   FE4C                    CP "L"
A936   201D                    JR NZ, ERROR
A938   CD87A9                  CALL SIGUIE
A93B   FE4C                    CP "L"
A93D   2016                    JR NZ, ERROR
A93F   1817                    JR ESCALL           ; para llegar aquí la
                                                   ; palabra ha de ser correcta
A941   FE46            FREE    CP "F"              ; lo mismo con !FRE
A943   2010                    JR NZ, ERROR
A945   CD87A9                  CALL SIGUIE
A948   FE52                    CP "R"
A94A   2009                    JR NZ, ERROR
A94C   CD87A9                  CALL SIGUIE
A94F   FE45                    CP "E"
A951   2002                    JR NZ, ERROR
A953   181A                    JR ESFRE            ; es !FRE
A955   C3F001          ERROR   JP 496              ; el 'VECTOR' original
A958   CD87A9          ESCALL  CALL SIGUIE
A95B   D7                      RST 16
A95C   821C                    DEFW 7298           ; evalúa la siguiente
                                                   ; expresión BASIC como
                                                   ; numérica, causando un error si no.
                                                   ; En tiempo de ejecución,
                                                   ; deja el valor en la
```

```
                                              ; pila del calculador.
A95E    CDB705              CALL 1463         ; mira si es el final
                                              ; de la sentencia y vuelve
A961    D7                  RST 16            ; en el tiempo de ejecución
A962    A22D                DEFW 11682        ; pila a BC, ver cap. 8
A964    38EF                JR C, ERROR       ; el acarreo es 1 si
                                              ; supera a 65535
A966 ED436BA9               LD (DEST), BC     ; la dirección del CALL
A96A    D7                  RST 16
A96B                DEST    DEFS 2            ; y la pone aquí
A96D    1815                JR FINAL          ; la ROM de 16K estará
                                              ; presente al llamar a la rutina
A96F    CDB705      ESFRE   CALL 1463         ; sale durante
                                              ; la comprobación de sintaxis
A972    210000              LD HL, 00         ; borra HL
A975    39                  ADD HL, SP        ; suma la dirección de SP
A976 ED5B655C               LD DE, (23653)    ; esta es STKEND
A97A    ED52                SBC HL, DE        ; resta de la dirección en SP
A97C    E5                  PUSH HL           ; para obtener el espacio
                                              ; libre para el BASIC
A97D    C1                  POP BC            ; el resultado ahora en BC
A97E    D7                  RST 16
A97F    2B2D                DEFW 11563        ; apila BC, ver cap. 8
A981    D7                  RST 16
A982    E32D                DEFW 11747        ; imprime el valor en la pila
A984    C3C105      FINAL   JP 1473           ; vuelve a BASIC
A987    D7          SIGUIE  RST 16
A988    2000                DEFW 32           ; rutina NEXT_CH de la ROM
A98A    E6DF                AND 223           ; que sea en mayúsculas
A98C    C9                  RET
```

Normalmente, los comandos o funciones de ampliación del BASIC terminarán con un salto al intérprete principal, como en el listado anterior, y es muy importante que sea la ROM de 8K la que esté conectada cuando se haga este salto, pues si no el programa fallará.

Para mayor información sobre este tema se recomienda la lectura de Programación del Interface I y Microdrive (Núñez Castain, Anaya Multimedia, 1985).


PUSH

8. El calculador

El Spectrum contiene un poderoso calculador en la ROM que puede ser usado en beneficio del programador en código máquina. Como tiene 66 rutinas diferentes, examinaré detalladamente sólo las más útiles. Para usar el calculador es importante entender la forma en la que el Spectrum maneja los números, cómo situarlos para que pueda cogerlos y cómo recuperar la respuesta de los cálculos terminados.

Todos los números usados por el calculador se almacenan como 5 bytes en representación de punto flotante binario o como pequeños enteros.

Representación de enteros pequeños

El primer byte es siempre 0.

El segundo byte es el signo 255 (FFh) para negativo o 0 para positivo.

El tercero y cuarto bytes son el número actual en el formato Z80 estándar, el byte bajo primero.

El byte final es siempre 0.

El número 0 se considera positivo.

Representación en punto flotante

El primer byte es el exponente; éste es el número de veces que el punto binario ha sido movido a la izquierda, para hacer que el bit 7 esté a 1 ya la derecha del punto binario; el bit 7 del exponente indica en qué dirección ha sido movido el punto binario. Si está a 1, el punto ha sido movido a la izquierda, como en el ejemplo siguiente.

Tomar el número 126 decimal, 7Eh, que en binario es 01111110. El punto binario, que debe ponerse, está en la derecha. Cuesta siete movimientos a la izquierda hacer que el bit más significativo esté a la derecha del punto binario.

El proceso es el siguiente:

0 movimientos 0 1 1 1 1 1 1 0.

1 movimientos 0 1 1 1 1 1 11 .0

2 movimientos 0 1 1 1 11.1 0

3 movimientos 0 1 1 1 1 . 1 1 0

y así hasta el último

7 movimientos 0. 1 1 1 1 1 1 0

Cualquiera de los bits a la izquierda del punto binario estará siempre a 0; por tanto, éste y el punto binario pueden ser descartados, dado que se conoce su posición para cualquier número. Normalmente, nosotros hacemos esto con números decimales, muchas veces inconscientemente, al no mostrar un punto decimal a la derecha de un número entero, aunque todos sabemos dónde está.

Este proceso nos da la parte del número conocida como la mantisa, que, para el ejemplo dado, es 11111100 en binario, y el exponente es 7 (el número de veces que el punto binario ha sido movido a la izquierda). El bit más significativo de la mantisa será siempre 1, por lo que este bit puede ser usado para mostrar el signo del número; 1 para un número negativo y 0 para uno positivo.

El exponente se expresa en binario con signo; el bit 7 está a 1, si el punto binario ha sido movido a la izquierda, ya 0 si ha sido movido a la derecha. De este modo, en el ejemplo anterior la mantisa era 7, en binario 00000111, pero el punto se había movido a la izquierda; por tanto, el bit 7 debe estar a 1, quedando 10000111 ó 135 decimal. Por consiguiente, el número expresado en cinco bytes queda:

	Exponente	Mantisa			
Binario	10000111	01111100	00000000	00000000	00000000
Decimal	135	124	0	0	0
Hexadecimal	87	7C	0	0	0

Normalmente, el número dado en el ejemplo anterior hubiera sido almacenado como entero pequeño, pero lo hemos usado aquí para simplificar.

El calculador usa su propia pila, en la que guarda todos los números sobre los que esté trabajando, y lo primero que se debe hacer antes de realizar cualquier cálculo es poner los números que van a utilizarse dentro de la pila del calculador. Esto se puede realizar de tres formas diferentes:

1. Un número puede ser guardado en la pila cogiéndolo de un registro o de una pareja de registros y usando una rutina de la ROM para convertirlo al formato del calculador.

2. El número se puede cambiar a la forma usada por el calculador, y entonces se guarda en la pila.

3. El número se puede escribir en la memoria en representación ASCII, y se puede usar el comprobador de sintaxis del BASIC para leerlo y situarlo dentro de la pila del calculador en la forma correcta.

Cada método tiene sus ventajas e inconvenientes, y cada uno se presta a diferentes tipos de números. Ahora consideraré el uso de cada uno de ellos.

1. Para números enteros pequeños hay dos rutinas que se pueden usar:

   CALL 11560 (2D28h)

   Esta se emplea poniendo el número que va a ser transferido a la pila del calculador dentro del registro A. Obviamente, el rango está limitado de 0 a 255 (0- FFh), y el número debe ser positivo.

   CALL 11563 (2D2Bh)

   Esta aceptará números en el rango de 0 a 65535 (0-FFFFh) almacenados en la pareja de registros BC. De nuevo el número sólo puede ser positivo, a menos que el inicio de la rutina sea evitado. La llamada se hace entonces a 11569 (2D31h), con el registro A puesto a 0 y el registro E puesto a 255 (FFh); entonces el número será transferido al calculador como negativo.

2. Para números en forma de cinco bytes similar a la del calculador se usa la rutina siguiente:

   CALL 10934 (2AB6h)

   La representación de cinco bytes del número, como se describió anteriormente en Representación en punto flotante, debe estar en los registros A, E, E, C, B; el exponente, en el registro A, y la mantisa, en los otros cuatro registros en orden.

3. Representación ASCII:

   CALL 11419 (2C9Bh)

   Siempre que el intérprete de BASIC se encuentra con un número en una línea BASIC que se está introduciendo, sitúa la forma binaria de cinco bytes del número después de la versión ASCII, lista para usar más tarde cuando el programa se ejecute. La rutina usada para hacer esto puede realizar también la conversión para un programa de código máquina escrito por el usuario. Esta rutina evita todos los problemas de conversión de números, bien manualmente o bien escribiendo una rutina en su propio programa que los convierta, y merece una detallada atención.

   Para usar esta rutina, la variable del sistema CH_ADD 23645 (5C5Dh) debe contener la dirección del primer carácter del número que se quiere guardar, y el registro A debe contener el código del carácter al que señala CH_ADD. El primer carácter será, normalmente, el dígito más significativo de un número decimal; pero, ya que la rutina es la que se usa para explorar las líneas del BASIC, podría ser el código binario 196 (C4h), si los dígitos siguientes son un número binario. Este caso sólo puede ser convertido a 16 dígitos, en decimal 65535 (FFFFh). Cualquier intento de superarlo se convertirá en un salto a la rutina de manejo de errores del BASIC. El formato con E (exponencial) puede ser usado si así se desea, y el número debe estar ordenado exactamente igual que si estuviera en una línea BASIC.

   Después de los caracteres que forman el número y, si se ha usado, el exponente, se debe añadir un byte con valor 13 (0Dh). Esto permite a la rutina saber que ha llegado al final del número y que no hay nada más que buscar.

Hay dos rutinas de la ROM que hacen lo opuesto a las dos rutinas de apilar A y apilar BC. Estas toman la última entrada en la pila del calculador y la convierten en un número entero, si es posible. Si el número es demasiado largo, entonces la bandera de acarreo estará a 1 a la salida, y si el número es negativo la bandera 0 se coloca a 0. Para un número positivo, la bandera 0 se colocará a 1. El número será borrado de la pila modificando el puntero, pero la pareja de registros DE seguirá señalándolo en la memoria, permitiendo recuperarlo si es necesario para alguna otra cosa, aunque es fácil duplicarlo antes de intentar la recuperación, y entonces borrar la copia si el cálculo sale bien. Las direcciones de llamada son:

PILA hacia A: CALL 11733 (2DD5H)

PILA hacia BC: CALL 11685 (2DA5H)

Existe también la rutina contraria a la expuesta anteriormente en 11249 (2BF 1h), la cual devuelve el último valor de la pila a los mismos registros. Esta rutina también borra el número de la pila, pero dado que no se puede recuperar el número, las banderas no se modifican. Debe hacerse un duplicado en la pila si se necesita tener el número en la misma después de la recuperación.

PILA hacia A, E, D, C, B: CALL 11249 (2BF 1 H)

Cuando los números son generados desde dentro del programa, la rutina para imprimir un número desde la pila puede ser usada para "imprimir" el número en un espacio de la memoria - lo opuesto a imprimirlo en pantalla- en la forma ASCII. El número máximo de posiciones de memoria que se necesitarán para un solo número es 14.

Esto se consigue escribiendo una subrutina que sitúa los contenidos del registro A dentro de la posición de la memoria siguiente cada vez que se le llama, abriendo un canal que señala a su propia subrutina, y haciendo este canal activo poniendo su dirección de base en CUR_CHL 23633 (5C51h). Se puede llamar entonces a la rutina de la ROM en 11747 (2DE3h) para imprimir el número. Su rutina será llamada entonces con cada carácter del número en secuencia, que debe guardar en las posiciones de la memoria, preparado para usar después. Recuerde que el índice que se usa para señalar a las direcciones donde se guarda el número no puede ser guardado en un registro entre llamadas a su rutina ni tampoco en la pila de la máquina; por tanto, se deben reservar dos bytes de memoria para salvar esta dirección. Un programa que haga esto puede ser algo así:

```
A2BB   21DFA2  10    MODIFI  LD HL, ESPACIO
A2BE   22EFA2  20            LD (ES_PA), HL
A2C1   2A515C  30            LD HL, (23633)     ; CUR-CHL
A2C4   E5      40            PUSH HL
A2C5   11D6A2  50            LD DE, INICIO      ; hace que su subrutina sea el
A2C8   4E      60            LD C, (HL)         ; canal actual y guarda la
A2C9   73      70            LD (HL), E         ; dirección original para
A2CA   23      80            INC HL             ; volverla a poner al final.
A2CB   46      90            LD B, (HL)
A2CC   72      100           LD (HL), D
A2CD   C5      110           PUSH BC
A2CE   CDE32D  120           CALL 11747         ; la rutina que imprime los números
A2D1   C1      130           POP BC             ; Devuelve la dirección original
A2D2   E1      140           POP HL             ; el número del canal de destino.
A2D3   71      150           LD (HL), C
A2D4   23      160           INC HL
A2D5   70      170           LD (HL), B
               180   ; EL RESTO DEL PROGRAMA VIENE AQUI
A2D6   2AEFA2  190   INICIO  LD HL, (ES_PA)
A2D9   77      200           LD (HL), A
A2DA   23      210           INC HL
A2DB   22EFA2  220           LD (ES_PA), HL
A2DE   C9      230           RET
A2DF           240   ESPACIO DEFS 16
A2EF   DFA2    250   ES_PA   DEFW ESPACIO
```

Esta es la rutina usada y comentada previamente. Cuando se la llama, toma la parte superior de la pila del calculador y la saca en ASCII, como un número decimal, por la corriente actual. El número se retira de la pila.

Siempre que se use el calculador, es importante asegurarse de que la pila está equilibrada. El calculador siempre obra de una forma prefijada (cuando se usa correctamente), y dará resultados erróneos cuando está desequilibrado. También se debe dejar espacio entre RAMTOP y STKBOT, para que la pila del calculador crezca hacia arriba y la pila de la máquina hacia abajo sin chocarse entre sí.

Dado que algunas rutinas del calculador lo usan recursivamente, será necesario dejar más espacio del que aparentemente usa. De hecho, es mejor pecar de exceso debido a este hecho. Si existe alguna duda de qué es lo que hay en la pila después de completar algunas operaciones, se puede vaciar con un CALL 5823 (16BFh), pero recuerde que esto borrará todo lo que hay en la pila.

Uso del calculador

Para entender cómo manejar el calculador, puede ayudarle a comprenderlo el considerarlo como un procesador separado, con su propio conjunto de instrucciones, que se conectan con un RST 40 (28h) y desconectan con un ENDCALC, código 56 (38h).

Durante el tiempo que el calculador está encendido, sus códigos se cogerán de las memorias siguientes a la RST 40. Estos son una serie de bytes definidos en un programa normal de Z80. Cuando el calculador encuentra una instrucción ENDCALC, el control es devuelto al Z80, y éste continúa la ejecución del programa desde la dirección siguiente a ENDCALC. Algunos códigos necesitan operandos, y otros sólo pueden utilizarse al comienzo de una serie, dado que requieren que los registros Z80 estén colocados de una forma especial para operar.

Cada vez que se usa la pila, su longitud varía en cinco bytes. Siempre que un número es situado en la pila se comprueba si hay espacio. Cuando el espacio no es suficiente, se producirá un error en el BASIC.

El calculador no está limitado sólo a operaciones numéricas; se usa también para producir una cadena de BASIC y funciones VAL. Estas las estudiaremos más tarde.

Los códigos que comprende el calculador se explican a continuación, con la descripción de su operación. En cada caso, el cambio de la pila del calculador se da en bytes -se usan cinco por cada valor- para la ejecución, por el calculador, de un solo código.

X es el valor por debajo de Y en la pila del calculador, siendo Y el último valor almacenado. El resultado de un cálculo se deja siempre en la parte superior de la pila, y este resultado es representado por Z. Por ejemplo, para el código de resta (03), si X = 5 e Y = 9, X se colocará dentro de la pila del calculador seguida por Y, la RST 40 (28h) estará seguida por un byte 03. Después de la operación, la pila del calculador contiene - 5, que es el resultado que hemos denominado Z. Además, X e Y serán borradas.

Los saltos se hacen desde la posición en la que se almacena el operando que indica la distancia según la forma estándar Z80.

Código	Función	Operación
00	SALTA SI CIERTO	Salta la distancia (en notación de complemento a 2) indicada en el operando (el byte definido después del código 00) y si y no es 0 (CAMBIO PILA - 5)
01	INTERCAMBIA	Invierte el orden de X e Y en la pila (CAMBIO PILA 0)
02	BORRAR	Quita Y de la pila. Z = X. (CAMBIO PILA - 5)
03	RESTAR	X - Y = Z (CAMBIO PILA - 5)
04	MUL TIPLICAR	X * Y = Z (CAMBIO PILA - 5)
05	DIVIDIR	X / Y = Z (CAMBIO PILA - 5)
06	ELEVAR A POTENCIA	X'Y = Z (CAMBIO PILA - 5)
07	OR BINARIO	X o Y. Si = 0 entonces Z = X; si no, Z = 1 (CAMBIO PILA - 5)

08	AND BINARIO	X e Y, ambos deben ser números. Si y = 0, entonces Z = 0; si no, Z = X. Hay un código distinto para manejar cadenas: 16 (10h). (CAMBIO PILA - 5)

La secuencia de códigos desde el 09 al 14 (0Eh) maneja valores numéricos. Cada código sólo puede ser usado como la primera operación después del RST 40 (28h), ya que el registro B debe contener el código en el momento de realizarse la operación.

Hay un segundo grupo de códigos para comparaciones de cadenas; todos vuelven con Z = 1, si la comparación da resultado correcto, o Z = 0, si es falsa. (CAMBIO PILA - 5)

Código			
09	Y < = X		
10 (0Ah)	Y > = X		
11 (0Bh)	Y < > X		
12 (0Ch)	Y > X		
13 (0Dh)	Y < X		
14 (0Eh)	Y = X		
15 (0FH)	ADICION	X + Y = Z	(CAMBIO PILA- 5)

En la siguiente secuencia de códigos, X y/o y deben contener los parámetros de una cadena. Los detalles de cómo conseguir estos parámetros y cómo ponerlos en la pila del calculador se dan posteriormente. La parte o partes que son parámetros de cadenas están marcadas con el símbolo $.

Código	Función	Operación
16 (10h)	$ Y n.º	X$ e Y. Si y = 0, entonces Z$ será una cadena vacía. (CAMBIO PILA = 5)

Los códigos del 17 al 22 (11h-16h) son los equivalentes para cadenas de los códigos 09-14 (0Eh) y son llamados con el registro B conteniendo el código. De nuevo Z = 1, la comparación es correcta, o Z = 0, si es falsa.

17 (11h)		Y$ < = X$	(CAMBIO PILA - 5)
18 (12h)		Y$ > = X$	(CAMBIO PILA - 5)
19 (13h)		Y$ < > X$	(CAMBIO PILA - 5)
20 (14h)		Y$ > X$	(CAMBIO PILA - 5)
21 (15h)		Y$ < X$	(CAMBIO PILA - 5)
22 (16h)		Y$ = X$	(CAMBIO PILA - 5)
23 (17h)	ADICCION	X$ + Y$ = Z$.	Las dos cadenas son unidas en el área de trabajo, y los nuevos parámetros se devuelven en Z$. Recuerde que si no hay espacio para copiar las dos cadenas dentro del área de trabajo se producirá un error del BASIC. (CAMBIO PILA - 5)

24 (18h)	VAL$	VAL$ Y$ = Z$.	La nueva cadena se crea en el área de trabajo de la misma forma que antes. El código debe estar contenido también en el registro B cuando se realiza la llamada. Cualquier tipo de error causará un error del BASIC. Esta es la rutina usada por el intérprete de BASIC, y está sujeta a todo el proceso de comprobación de sintaxis (CAMBIO PILA 0)
25 (19h)	USR$	Z = USR$.	Este también es usado por el intérprete de BASIC, y está sujeto a la comprobación de sintaxis. Y$ debe contener los parámetros de una variable alfanumérica (terminada en $), que contenga una sola letra desde la A hasta la U. Al terminar, Z será la dirección del gráfico definible por el usuario. (CAMBIO PILA 0)
26 (1Ah)	LEER EN		Esta rutina permite poner un solo byte en el AREA DE TRABAJO mediante cualquier corriente de la 0 a la 15 (0Fh), según se le indique en Y. El byte es trata como si fuese una cadena, y Z$ son los parámetros de esta cadena. Si la bandera de acarreo no está puesta a 1 por la rutina de entrada, no se realiza ninguna acción y Z$ es devuelta como una cadena nula.
27 (1Bh)	NEGAR	Z = Y	Pero con el signo cambiado. (CAMBIO PILA 0)
28 (1Ch)	CODE	Z = CODE Y$.	Tal como lo usa el intérprete BASIC. (CAMBIO PILA 0)
29 (1Dh)	VAL	Z = ZAL Y$.	Tal como lo usa el intérprete BASIC y en realidad, accede al comprobador de líneas para dar el resultado en Z. Sujeto a los errores del BASIC. (CAMBIO PILA 0)
30 (1Eh)	LEN	Z = LEN Y$.	Esta es más fácil de hacer sin usar el calculador; todo lo que se hace es apilar los bytes de tamaño de los parámetros de la cadena. (CAMBIO PILA 0)

Todas las funciones algebraicas siguientes devuelven la respuesta (Z) en la parte superior de la pila del calculador y el tamaño de ésta no cambia.

Código Operación

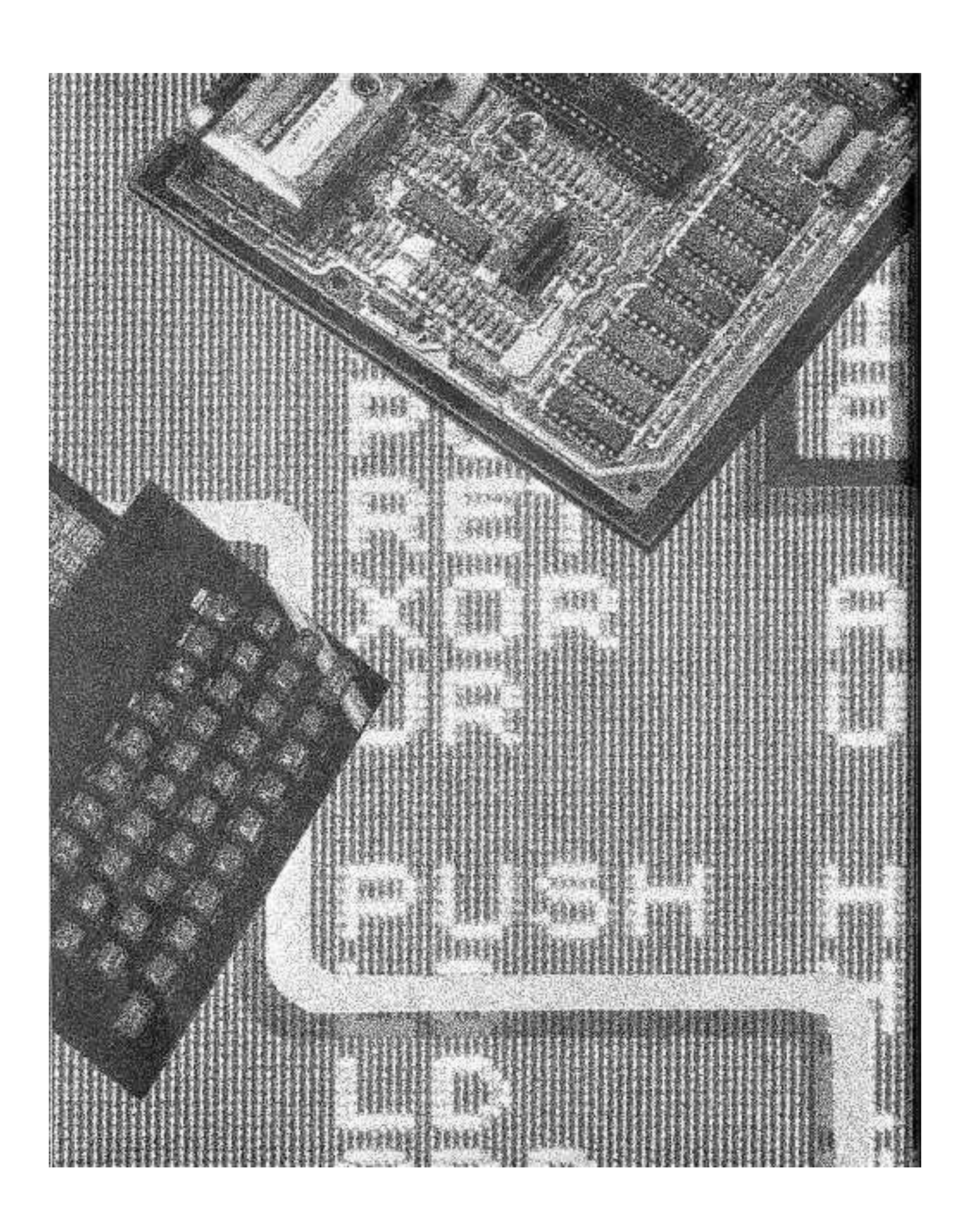
31 (1Fh)	Z = SIN Y	
32 (20h)	Z = COS Y	
33 (21h)	Z = TAN Y	
34 (22h)	Z = ASN Y	
35 (23h)	Z = ACS Y	
36 (24h)	Z = ATN Y	
37 (25h)	Z = LN Y	
38 (26h)	Z = EXP Y	
39 (27h)	Z = INT Y	
40 (28h)	Z = SQR y	
41 (29h)	Z = SGN y	
42 (2Ah)	Z = ABS Y	
43 (2Bh)	PEEK	Z = PEEK Y. Tal como lo usa el intérprete BASIC. (CAMBIO PILA 0)
44 (2Ch)	IN	Z = IN Y. Esta ejecuta la instrucción del Z80 IN A, (C) después de pasar el Y de la pila a la pareja de registros BC como un número entero. (CAMBIO PILA 0)
45 (2Dh)	USR No.	Precaución. Esta producirá la ejecución de un salto a la dirección Y. Es usada por el BASIC para saltar al código máquina. La dirección de retorno será 11563 (2D2Bh); la rutina pasará el valor en el registro BC a la pila al serle devuelto el control. Este es un código interesante, dado que abre la posibilidad de usar rutinas de la ROM y la RAM recursivamente desde dentro del calculador. (CAMBIO PILA 0)
46 (2Eh)	STR$	Z$ = Y. El valor superior de la pila del calculador se imprime en el AREA DE TRABAJO y se evalúa como una cadena; sus parámetros se ponen en la pila del calculador. (CAMBIO PILA 0)
47 (2Fh)	CHR$	Z$ = CHR$ Y. Si 0 < Y < 255, entonces se crea un espacio en el AREA DE TRABAJO, y el valor es transferido allí como un byte. Este se interpreta entonces como una cadena, y sus parámetros se devuelven en Z$. (CAMBIO PILA 0)
48 (30h)	NOT	Z = 1, si Y = 0; si no, Z = 0. (CAMBIO PILA 0)
49 (31h)	DUPLICAR	Z = Y. Y es duplicada. (CAMBIO PILA + 5)

50 (32h)	X MOD Y	Esta devuelve Z = INT (X/Y) y debajo de Z (donde X estaba en un principio) X - INT (X/Y). (CAMBIO PILA 0)
51 (33h)	SALTO	Esta lo único que hace es el equivalente de una instrucción del Z80 JR (salto relativo), pero para el calculador. La longitud del salto se toma de la posición siguiente al código. La pila del calculador no es modificada. (CAMBIO PILA 0)
52 (34h)	STK DATA	Esta permite leer un valor de las posiciones siguientes al código. Los bits 6 y 7 del primer byte (el exponente) que está a continuación del código determinan el número de bytes que forman la mantisa, 00 BIN, para 1, al 11 BIN, para 4. Entonces se almacena el número en punto flotante en la pila del calculador. (CAMBIO PILA + 5)
53 (35h)	DEC JR NZ	Esta es una equivalente directa del código del Z80 DJNZ, pero para el calculador. Se toma la variable del sistema BREG como si fuera el registro B. No es segura para usar, dado que esta rutina es usada ampliamente por el calculador para sus propios fines, y el valor de BREG puede variar. (CAMBIO PILA 0)
54 (36h)	Y < 0	Z = 1, si Y < 0; si no, Z = 0. (CAMBIO PILA 0)
55 (37h)	Y > 0	Z = 1, si Y > 0; si no, Z = 0. (CAMBIO PILA 0)
56 (38h)	ENDCALC	Devuelve el control al Z80 en la siguiente dirección. (CAMBIO PILA 0)
57 (39h)	COGE ARGT	Usada internamente por el calculador para establecer el valor de Y (qué para el fin de Z como resultado, será Z) en SIN Y o COS Y. (CAMBIO PILA 0)
58 (3Ah)	TRUNCAR	Z = INT Y, donde Z es Y truncada hacia 0. (CAMBIO PILA 0)
59 (3Bh)	FP CALC 2	Usado por el intérprete para ejecutar una sola operación del calculador. No tiene uso práctico para los fines de este libro.
60 (3CH)	E A FP	Z = Y * (10 elevado al registro A). (CAMBIO PILA 0)
61 (3Dh)	RE APILA	Z = la versión en punto flotante de Y, donde Y puede ser un número entero pequeño. (CAMBIO PILA 0)
62 (3Eh)	SERIES	Usado internamente por el calculador para generar un polinomio de Chebyshev. Prácticamente no se usa, ya que es llamado por las rutinas que lo necesitan.
63 (3Fh)	APILAR No.	Como antes, pero usado para apilar constantes.
64 (40h)	ST MEM	Usado para almacenar en el área de la memoria. A la

		entrada, el registro A debe contener un valor comprendido entre C0 y C5, según cuál de las cinco posiciones de la memoria se va a usar. (CAMBIO PILA - 5)
65 (41h)	REC MEM	Recuperación de memoria. Es la inversa de la rutina anterior. (CAMBIO PILA + 5)

Para las operaciones de cadenas, los parámetros pueden ser pasados a la pila del calculador mediante la rutina en 10934 (2AB6h) ya mencionada. La pareja de registros BC contiene la longitud de la cadena; la dirección de inicio está en la pareja de registros DE, y el nombre del registro en A (en la forma comentada en el capítulo 2 en Salvando, cargando y verificando).

Para comprender mejor el calculador, encontrará en el apéndice G, Subrutinas más útiles, una rutina que le permitirá hacer experimentos con él.


PUSH

Apéndice A Conversión de hexadecimal a decimal

MSB

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	256	512	768	1024	1280	1536	1792	2048	2304	2560	2816	3072	3328	3584	3840
1	4096	4352	4608	4864	5120	5376	5632	5888	6144	6400	6656	6912	7168	7424	7680	7936
2	8192	8448	8704	8960	9216	9472	9728	9984	10240	10496	I0752	11008	11264	11520	1l776	12032
3	12288	12544	12800	13056	13312	13568	13824	14080	14336	14592	14848	15104	15360	15616	15872	16128
4	16384	1664O	16896	17152	17408	17664	17920	18176	18432	18688	18944	19200	19456	19712	19968	20224
5	20480	20736	20992	21248	21504	21760	22016	22272	22528	22784	23040	23296	23552	23808	24064	24320
6	24576	24832	25088	25344	25600	25856	26112	26368	26624	26880	27136	27392	27648	27904	28l6O	28416
7	28672	28928	29184	29440	29696	29952	30208	30464	30720	30976	31232	31488	31744	32000	32256	32512
8	32768	33024	33280	33536	33792	34048	34304	34560	348l6	35072	35328	35584	35840	36096	36352	36608
9	36864	37120	37376	37632	37898	38144	38400	38656	38912	39168	39424	39680	39936	40192	40448	40704
A	40960	41216	41472	41728	41984	42240	42496	42752	43008	43264	43520	43776	44032	44288	44544	44800
B	45056	45312	45568	45824	46080	46336	46592	46848	47104	47360	476l6	47872	48128	48384	48640	48896
C	49152	49408	49664	49920	50176	50432	50688	50944	51200	51456	51712	51968	52224	52490	52736	52992
D	53248	53504	53760	540l6	54272	54528	54784	55040	55296	55552	55808	56064	56320	56576	56932	57088
E	57344	57600	57856	58112	58368	58624	58880	59136	59392	59648	59904	60l60	604l6	60672	60928	61184
F	61440	61696	61952	62208	62464	62720	62976	63232	63488	63744	64000	64256	64512	64768	65024	65280

LSB

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	11	10	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	41
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	61	6B	69	10	71	72	13	74	75	16	71	18	19
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	22	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	l59
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	176	177	178	79	180	181	182	183	184	185	186	187	189	189	190	191
C	192	193	194	195	196	191	198	199	200	201	202	203	204	205	206	201
D	208	209	210	211	212	213	214	215	210	211	218	219	220	221	222	223
E	224	225	226	221	228	229	230	231	232	233	234	235	236	231	238	239
F	240	241	242	243	244	245	246	241	248	249	250	251	252	253	254	255

Nibbles

HEX	DEC	BIN	HEX	DEC	BIN
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	A	10	1010
3	3	0011	B	11	1011
4	4	0100	C	12	1100
5	5	0101	D	13	1101
6	6	0110	E	14	1110
7	7	0111	F	15	1111

Apéndice B Mapa de memoria del Spectrum

VARIABLES

FIJAS

PRAMT FFFFh o 7FFFh

65535 o 32767

(23675) UDG
(23730) RAMTOP
(23653) STK END
PILA DEL CALCULADOR
(23651) STK BOT
AREA DE TRABAJO
(23649) WORKSP
AREA DE EDICION
(23641) E LINE
VARIABLES
(23627) VARS
BASIC
(23635) PROG
INFOR. CANALES
(23631) CHANS
MAPAS MICRODR

5CEFh 23731 ULTIMO BYTE DE LAS VARIABLES DE 8K

5CB6h 23733 ULTIMO BYTE DE LAS VARIABLES DE 16K

5BFFh 23551 ULTIMO BYTE DE LA MEMORIA INTERMEDIA DE LA IMPRESORA

5AFFh 23295 ULTIMO BYTE DEL MAPA DE ATRIBUTOS DE LA PANTALLA

57FFh 22527 ULTIMO BYTE DEL MAPA DE PANTALLA

3FFFh 16383 ULTIMO BYTE DE LA ROM DE 16K

Apéndice C Mapa de pantalla del Spectrum

La pantalla del Spectrum está dividida en 192 filas de 256 puntos. Cada línea contiene 32 bytes de memoria en orden ascendente. A continuación, se lista las direcciones del primer byte de cada línea de pantalla con la dirección del atributo que controla a éste. La línea 1 representa la línea superior en la pantalla.

LINEA	PIXEL	ATRIBUTO			
1	4000H	5800H	25	4060H	5860H
2	4100H		26	4160H	
3	4200H		27	4260H	
4	4300H		28	4360H	
5	4400H		29	4460H	
6	4500H		30	4560H	
7	4600H		31	4660H	
8	4700H		32	4760H	
9	4020H	5820H	33	4080H	5880H
10	4120H		34	4180H	
11	4220H		35	4280H	
12	4320H		36	4380H	
13	4420H		37	4480H	
14	4520H		38	4580H	
15	4620H		39	4680H	
16	4720H		40	4780H	
17	4040H	5840H	41	40A0H	58A0H
18	4140H		42	41A0H	
19	4240H		43	42A0H	
20	4340H		44	43A0H	
21	4440H		45	44A0H	
22	4540H		46	45A0H	
23	4640H		47	46A0H	
24	4740H		48	48A0H	
49	40C0H	58C0H	121	48E0H	59E0H
50	41C0H		122	49E0H	
51	42C0H		123	4AE0H	
52	43C0H		124	4BE0H	
53	44C0H		125	4CE0H	
54	45C0H		126	4DE0H	
55	46C0H		127	4EE0H	
56	47C0H		128	4FE0H	
57	40E0H	58E0H	129	5000H	5A00H
58	41E0H		130	5100H	
59	42E0H		131	5200H	
60	43E0H		132	5300H	
61	44E0H		133	5400H	
62	45E0H		134	5500H	
63	46E0H		135	5600H	
64	47E0H		136	5700H	
65	4800H	5900H	137	5020H	5A20H
66	4900H		138	5120H	
67	4A00H		139	5220H	
68	4B00H		140	5320H	
69	4C00H		141	5420H	
70	4D00H		142	5520H	
71	4E00H		143	5620H	
72	4F00H		144	5720H	
73	4820H	5920H	145	5040H	5A40H
74	4920H		146	5t40H	
75	4A20H		147	5240H	
76	4B20H		148	5340H	
77	4C20H		149	5440H	
78	4D20H		150	5540H	
79	4E20H		151	5640H	
80	4F20H		152	5740H	
81	4840H	5940H	153	5060H	5A60H
82	4940H		154	51b0H	
83	4A40H		155	5260H	
84	4840H		156	53b0H	
85	4C40H		157	5460H	
86	4D40H		158	5560H	
87	4E40H		159	5660H	
88	4F40H		160	5760H	
89	4860H	5960H	161	5080H	5A80H

90	4960H		162	5180H	
91	4A60H		163	5280H	
92	4B60H		164	5380H	
93	4C60H		165	5480H	
94	4Db0H		t66	5580H	
95	4E60H		167	5680H	
96	4F60H		168	5780H	
97	4880H	5980H	169	50A0H	5AA0H
98	4980H		170	51A0H	
99	4A80H		171	52A0H	
100	4880H		172	53A0H	
101	4C80H		173	54A0H	
102	4D80H		174	55A0H	
103	4E80H		175	56A0H	
104	4F80H		176	57A0H	
105	48A0H	59A0H	177	50C0H	5AC0H
106	49A0H		178	51C0H	
107	4AA0H		179	52C0H	
108	48A0H		180	53C0H	
109	4CA0H		181	54C0H	
110	4DA0H		182	55C0H	
111	4EA0H		183	56C0H	
112	4FA0H		184	57C0H	
113	48C0H	59C0H	185	50E0H	5AE0H
114	49C0H		186	51E0H	
115	4AC0H		187	52E0H	
116	4BC0H		188	53E0H	
117	4CC0H		189	54E0H	
118	4DC0H		190	55E0H	
119	4EC0H		191	56E0H	
120	4FC0H		192	57E0H	

Apéndice D El mapa del teclado

	D4	D3	D2	D1	D0
254 (FEh)	V	C	X	Z	C/S
253 (FDh)	G	F	D	S	A
251 (FBh)	T	R	E	W	Q
247 (F7h)	5	4	3	2	1
239 (EFh)	6	7	8	9	0
223 (DFh)	Y	U	I	O	P
191 (BFh)	H	J	K	L	ENT
127 (7Fh)	B	N	M	S/S	B/S

Apéndice E El juego de caracteres del Spectrum

Código	Carácter	Notas especiales
0 y 1	Nada	Usados sólo después de INK, PAPER, OVER, INVERSE, FLASH, BRIGHT, AT o TAB.
2-5	Nada	Usados sólo después de INK, PAPER, AT o TAB.
6	TAB	Hace una tabulación (TAB) a la siguiente posición de media pantalla.
7	EDIT	Código devuelto por la rutina de entrada del teclado si se pulsan CAPS SHIFT y 1. No imprimible. Frecuentemente el código BELL en las impresoras y terminales.
8	RETROCESO	Código devuelto por la rutina de entrada del teclado si se pulsan CAPS SHIFT y '5'. Se puede imprimir en la pantalla para retroceder un espacio. También es reconocido por muchas impresoras, ya que es el código ASCII de retroceso.
9	AVANCE	Código devuelto por la rutina de entrada del teclado si se pulsan CAPS SHIFT y '8'. No se puede usar para mover un espacio a la derecha en la pantalla, ya que el Spectrum no actualiza las posiciones de pantalla después de su uso. Código ASCII correspondiente al TAB.
10	ABAJO	Como el anterior, pero con CAPS SHIFT y '6'. Es el código ASCII de salto de línea.
11	ARRIBA	Como el anterior, pero con CAPS SHIFT y '7'. Código ASCII de tabulación vertical.
12	BORRAR	Como el anterior, pero con CAPS SHIFT y 'O'. Salto de hoja en ASCII.
13	ENTER	Código devuelto cuando se pulsa la tecla ENTER; ejecuta un retorno de carro (vuelve a la posición izquierda) y salto de línea cuando se imprime en la pantalla. Es también el código ASCII para el retorno de carro.
14		Precede a un número en una línea de un programa BASIC. No se usa prácticamente. El código SO en ASCII.
15		No se usa en el Spectrum. Código SI en ASCII.
16		Código de control de la tinta usado para preceder al número del color de la tinta (INK). Por ejemplo, para cambiar a rojo todos los caracteres siguientes que se impriman en pantalla, deberá usar la rutina RST 16 con el registro A conteniendo un 16 seguido por 2. Nótese que no es el carácter ASCII '2', sino el valor 2. Esto se demuestra en el programa DeBASE.
17		Como el anterior, pero para el papel (PAPER).
18		Igual al anterior, para parpadeo (FLASH); el código siguiente puede ser sólo 0 ó 1.

19	Como el anterior, para brillo (BRIGHT).
20	Como el anterior, para inverso (INVERSE).
21	Como el anterior, para OVER.
22	Código de control AT; debe estar seguido por los valores de la línea y de la columna.
23	Como el anterior, pero con TAB ; sólo necesita el valor de la columna.

Los códigos del 24 al 31 no son usados por el Spectrum; sin embargo, el código 27 es el código escape usado por muchos periféricos, seguido por una letra, para indicar la acción que se pide. Mientras que el CHR$ 27 ha sido estandarizado como el código escape, no ha sucedido así con los códigos que le siguen.

Los códigos que quedan son todos representaciones de caracteres, con los códigos del 32 al 126 formando el conjunto ASCII estándar y que se muestran en el manual del Spectrum.

El código 127 --el símbolo copyright (©)- es el utilizado para borrar caracteres del ASCII. ¡Tenga cuidado!

Apéndice F Vectores de interrupción de la ROM

	16K ROM Mk 2 Spectrum	8K ROM Mk 1 Interface
I = 0	204300	23755
I = 1	52818	8501
I = 2	22269	25888
I = 3	39020	4196
I = 4	10419	52486
I = 5	2294	6701
I = 6	29149	51711
I = 7	16039	32459
I = 8	2088	14353
I = 9	65129	58170
I = 10	32802	1200
I = 11	58888	2039
I = 12	53183	59861
I = 13	52503	8205
I = 14	14367	49921
I = 15	27928	58884
I = 16	51984	32742
I = 17	8729	32477
I = 18	52481	4831
I = 19	49749	21965
I = 20	25705	56790
I = 21	51673	45182
I = 22	51568	65535
I = 23	12493	57851
I = 24	15582	61947
I = 25	23842	60952
I = 26	13824	52189
I = 27	7306	16129
I = 28	49947	6400
I = 29	2344	4870
I = 30	26573	65535
I = 31	3360	57855
I = 32	52513	23755
I = 33	33485	8501
I = 34	544	2588B
I = 35	49537	4196
I = 36	8527	52486
I = 37	23670	6701
I = 38	20'444	51711
I = 39	288	32459
I = 40	32348	14753
I = 41	58154	58170
I = 42	19754	1200
I = 43	23653	2039
I = 44	7117	59B61
I = 45	55781	8205
I = 46	23713	49921
I = 47	4569	58B84
I = 48	60208	32742

	16K ROM Mk 2 Spectrum	8K ROM Mk 1 Interface
I = 49	576401	32477
I = 50	13627	4831
I = 51	13256	21965
I = 52	1560	567910
I= 53	57124	45182
I = 54	34307	65535
I = 55	41231	57851
I = 56	65535	61947
I = 57	65535	60952
I = 58	65575	52189
I = 59	65535	16129
I = 60	255	6400
I = 61	0	4870
I = 62	255	65535
I = 63	60	255

Apéndice G Subrutinas útiles

Rutinas del calculador

Para una mejor comprensión y uso del calculador, las rutinas siguientes le permitirán experimentar con él. El fin de la primera rutina es el de la demostración:

```
A52F   D9        10            EXX             ; siempre debe salvar H'L' si
A530   E5        20            PUSH HL         ; quiere volver correctamente,
A531   D9        30            EXX
A532   CD32A5    40            CALL IMP_PI     ; imprime el inicio y fin de la pila.
A535   0135A5    50            LD BC, X        ; primer numero
A538   C5        60            PUSH BC         ; lo salva
A539   CD2B2D    70            CALL #2D2B      ; mete X en la pila
A53C   C1        80            POP BC          ; recupera X
A53D   CD2B2D    90            CALL #2D2B      ; lo mete de nuevo
A540   CDE32D    100           CALL #2DE3      ; imprime X
A543   3E20      110           LD A, #20
A545   D7        120           RST #10         ; imprime un espacio
A546   215EA5    130           LD HL, NUMERO   ; dirección de inicio del
                 140                           ; número en formato ASCII
A549   CD49A5    150           CALL APILA      ; lo guarda en la pila (será Y)
A54C   EF        160           RST #28         ; llama al calculador
A54D   31        170           DEFB #3         ; duplica Y
A54E   38        180           DEFB #38        ; desconecta el calculador
A54F   CDE32D    190           CALL #2DE3      ; imprime  Y
A552   EF        200           RST #28         ; llama al calculador
                 210   ; AQUI PUEDE INSERTAR UNA SERIE DE
                 220   ; BYTES PARA EXPERIMENTAR CON
                 230   ; EL CALCULADOR.
A553   38        240           DEFB #38        ; lo desconecta
A554   COE32D    250           CALL #2DE3      ; imprime el resultado
A557   C032A5    260           CALL IMP_PI     ; imprime el inicio y el fin de
                 270                           ; la pila para ver si esta
                 280                           ; bien.
A55A   D9        290           EXX             ; recupera H'L'
A55B   E1        300           POP HL
A55C   D9        310           EXX
A55D   C9        320           RET
A55E   31323334  330   NUMERO  DEFM "1234.567"
A566   0D        340           DEFB 13         ; siempre debe haber un
                 350                           ; después del número para
                 360                           ; indicar que se ha terminado.
```

Esta rutina apilará un número en formato ASCII:

```
A210   ED4B635C  10    IMP_PI  LD BC, (23651)  ; esta es la base de la pila
A214   CD2B2D    20            CALL #2D2B      ; la guarda en la pila
A217   3E42      30            LD A, "B"
A219   D7        40            RST #10
A21A   3E20      50            LD A, " "
A21C   07        60            RST #10
A21D   CDE32D    70            CALL #2DE3      ; imprime la dirección de la base
A220   3E20      80            LD A, " "
A222   D7        90            RST #10
A223   3E53      100           LD A, "8"
A225   D7        110           RST #10
A226   3E20      120           LD A," “
A228   D7        130           RST #10
A229   ED4B655C  140           LD BC, (23653)  ; esta es la parte
                 145                           ; superior de la pila
A22D   CD2B2D    150           CALL #2D2B
A230   CDE32D    160           CALL #2DE3      ; imprime la dirección
                 165                           ; de la parte superior
A233   3E0D      170           LD A, #0D
A235   D7        180           RST #10
A236   C9        190           RET
```

Esta rutina apilará un número en formato ASCII:

```
                        ; al llamar a esta rutina, el registro HL
                        ; debe contener la dirección de inicio
                        ; del número a meter en la pila.
A205  ED5B5D5C          APILA   LD DE, (23645)    ; esta es CH_ADD
A209  D5                        PUSH DE           ; la salva
A20A  225D5C                    LD (23645), HL    ; guarda en CH_ADD la dirección
                                                  ; del número
A20D  7E                        LD A, (HL)        ; pone el primer carácter en A
A20E  CD9B2C                    CALL #2C9B        ; guarda el numero en la pila
A211  D1                        POP DE            ; devuelve
A212  ED535D5C                  LD (23645), DE    ; el valor original a CH_ADD
A216  C9                        RET
```

Rutinas de manejo de los interfaces Morex y Kempston

A la entrada de estas subrutinas, el código ASCII, que se quiere sacar por el Centronics, debe estar en el registro A. No se expandirá ningún código; ver la sección Expandiendo claves para la salida en el capítulo 2, si necesita hacerlo.

Esta es la rutina de salida por el interfaz Centronics de Morex.

```
A1D3  F5                        PUSH AF           ; salva el código del carácter
A1D4  DBFB              OCUPAD  IN A, (#OFB)      ; lee la línea de ocupado de la impresora
A1D6  E601                      AND 1;            comprueba el bit 0
A1D8  20FA                      JR NZ, OCUPADA    ; si está a 1, la impresora esta
                                                  ; ocupada.
A1DA  F1                        POP AF            ; recupera el carácter
A1DB  D3FB                      OUT (#OFB), A     ; lo envía
A1DD  3E01                      LD A, 1           ; envía marca de validez
A1DF  D37F                      OUT (#7F), A
A1E1  AF                        XOR A
A1E2  D37F                      OUT (#7F), A
A1E4  C9                        RET               ; fin
```

Esta es la rutina de salida por el interfaz Centronics de Kempston :

```
A342  C5                        PUSH BC           ; salva los registros BC
                                                  ; ya que se usan para entrada/salida
A343  F5                        PUSH AF           ; salva el código del carácter
A344  01BFE2                    LD BC, #0E2BF     ; puerto de la línea de ocupado
A347  ED78              OCUPAD  IN A, (C)         ; lee la línea de ocupado
A349  1F                        RRA               ; pasa el bit cero a la bandera de acarreo
A34A  38FB                      JR C, OCUPADO     ; si está a 1 la impresora está ocupada
A34C  F1                        POP AF            ; recupera el carácter
A34D  05                        DEC B             ; cambia el puerto a 02BFh
A34E  05                        DEC B
A34F  ED79                      OUT (C), A        ; envía el carácter
A351  3EOE                      LD A, #OE         ; marca de validez a enviar
A353  06E3                      LD B, #OE3        ; cambia el puerto a E3BFh, puerto de validez
A355  ED79                      OUT (C), A        ; envía la marca
A357  3C                        INC A
A358  ED79                      OUT (C), A
A35A  C1                        POP BC            ; recupera el par BC
A35B  C9                        RET               ; fin
```

Rutina de sprites basada en interrupciones

```
                        ; programa de demostración de sprites
                        ; ver Capitulo 6
C8FF                            ORG 51455
C8FF                            ENT 51455

                        ; 51455 contiene el vector de interrupciones
                        ;
C8FF  2CC9                      DEFW 51500
                        ;
C901  3EC8              INICIA  LD A, 200
C903  ED47                      LD I, A
C905  ED4B83C9                  LD BC, (COORD)
```

```
C909    CD87C9    130             CALL DIBUJA
C90C    ED5E      140             IM 2
C90E    C9        150             RET
                  160
                  170     ; este es el programa real
                  180     ; que empieza en el vector de interrupciones
                  190
                  200
C92C              210             ORG 51500
                  220     ;
                  230     ; primero se salvan los registros a usar
                  240     ;
C92C    E5        250             PUSH HL
C92D    C5        260             PUSH BC
C92E    D5        270             PUSH DE
C92F    F5        280             PUSH AF
                  290
                  300     ; Salva TV_FLAG ya que puede ser alterada después
                  310     ;
C930    3A3C5C    320             LD A, (23612)
C933    F5        330             PUSH AF
                  340
                  350     ; salva CO_ORDS ya que será cambiada y debe ser
                  360     ; recuperada antes de volver
                  370     ;
C934    2A7D5C    380             LD HL, (23677)
C937    E5        390             PUSH HL
C938    ED4B83C9  400             LD BC, (COORD)              ; las coordenadas de los sprites
C93C    C5        410             PUSH BC                     ; las salva
                  420     ; llama a la rutina que dibuja la
                  430     ; sprite y como el dibujo se hace
                  435     ; con OVER 1, se borrará
                  440     ; la posición anterior
C93D    CD87C9    450             CALL DIBUJA
C940    C1        460             POP BC                      ; recupera las coordenadas de la sprite
                  470     ;
                  480     ; L es usada como un registro de banderas
                  490     ; para ver hacia donde se mueve la sprite
                  500     ; si el bit cero esta alto se mueve hacia arriba,
                  505     ; a cero indica hacia abajo.
                  510     ; bit 1 alto para izquierda,
                  515     ; bajo para derecha.
                  520     ;
                  530     ;
C941    2A85C9    540             LD HL, (BANDERA)
C944    CB45      550             BIT 0, L
C946    2807      560             RET Z, ARRIBA
C948    05        570     ABAJO   DEC B                       ; mueve la sprite hacia
                  575                                         ; abajo I puntos.
C949    200C      580             JR NZ, IZQUIE               ; si la bandera cero esta puesta
C94B    CB85      590             RES O, L                    ; ha llegado a la parte inferior de la
C94D    1808      600             JR IZQUIE                   ; pantalla y se cambia la dirección
C94F    04        610     ARRIBA  INC     B                   ; hacia arriba un punto
C950    78        620             LD      A, B                ; ¿ha alcanzado la parte superior?
C953    2002      640             JR      NZ, IZQUIE
C955    CBC5      650             SET     0, L                ; si es así, cambia la dirección
C957    CB4D      660     IZQUIE  BIT     1, L                ; lo mismo, pero para izquierda/derecha
C959    2807      670             JR      Z, DERECH
C95B    0D        680             DEC     C
C95C    200C      690             JR      NZ, CDIBUJ
C95E    CB8D      700             RES     1, L
C960    1808      710             JR      CDIBUJ
C962    0C        720     DERECH  INC     C
C963    79        730             LD      A, C
C964    FEFE      740             CP      254
C966    2002      750             JR      NZ, CDIBUJ
C968    CBCD      760             SET     1, L
                  770
                  780     ; se salvan las nuevas
                  785     ; coordenadas de la sprite
                  790     ;
C96A    ED4383C9  800     CDIBUJ  LD (COORD), BC              ; así como las banderas
C96E    2285C9    810             LD (BANDERA), HL
                  820
C971    CD87C9    830             CALL DIBUJA                 ; dibuja la nueva posición de la sprite
                  840     ;
```

```
                    850     ;
                    860     ; ya se ha realizado todo el dibujo por lo que
                    870     ; se devuelve su valor a las variables del sistema
                    880     ;
                    890     ;
C974    E1          900     NRECU   POP HL
C975    227D5C      910             LD (23677), HL
C978    F1          920             POP AF
C979    323C5C      930             LD (23612), A
                    940
                    950     ; se recuperan ahora los registros
C97C    F1          960             POP AF
C97D    D1          970             POP DE
C97E    C1          980             POP BC
C97F    E1          990             POP HL
                    1000
                    1010    ; se explora el teclado ya que no se había hecho antes
                    1020    ; debido a que las interrupciones estaban relocalizadas
                    1030    ;
C980    FF          1040            RST #38
C981    ED4D        1050            RETI
C983    0000        1060    COORD   DEFW 0
C985    0000        1070    BANDER  DEFW 0
                    1080    ;
                    1090    ; se va a usar D como contador,
                    1100    ; ya que se está usando B
                    1110    ;
C987    1604        1120    DIBUJA  LD D, 4
                    1130    ;
                    1140    ;
                    1150    ; salva las coordenadas ya que las rutinas
                    1155    ; de ROM las destruyen
                    1160    ;
                    1170    ;
                    1180    ;
C989    C5          1190            PUSH BC
C98A    7A          1200    BUCLE   LD A, D
C98B    D5          1210            PUSH DE                 ; salva el contador
                    1220
                    1230    ; el dibujo es hecho cuatro veces para crear una
                    1240    ; sprite más grande
C98C    FE04        1250            CP 4
C98E    280F        1260            JR Z, BUCLE4
C990    FE03        1270            CP 3
C992    2001        1280            JR NZ, BUCLE1
C994    0C          1290            INC C
C995    FE02        1300    BUCLE1  CP 2
C997    2001        1310            INC B
C99A    FE01        1330    BUCLE2  CP 1
C99C    2001        1340            JR NZ, BUCLE4
C99E    0D          1350            DEC C
                    1360
                    1370    ; se comprueba ahora que ROM está presente
                    1380    ;
C99F    3A1400      1390    BUCLE4  LD A, (#14)             ; coge una posición de memoria
                    1400                                    ; que tiene un valor distinto en
                    1410                                    ; cada ROM
C9A2    FDCB0286    1420            RES 0, (IY+2)           ; indica los colores de la pantalla
C9A6    C5          1430            PUSH BC
C9A7    FED5        1440            CP #D5
C9A9    28OC        1450            JR Z, ROM2              ; la ROM del interfaz 1 esta puesta
                    1460                                    ; de modo que no se llama a la rutina
                    1470                                    ; de la ROM principal directamente,
                    1480                                    ; sino que se debe activar antes.
                    1490
C9AB    CD4D0D      1500            CALL#D4D                ; coge los colores con los que dibujar
C9AE    FDCB57C6    1510            SET 0, (IY+87)          ; activa OVER 1 con la bandera P
C9B2    CDE522      1520            CALL 8933               ; dibuja
C9B5    1808        1530            JR CONT1                ; se salta el cambio de la ROM de 8K
                    1540                                    ; a la de 16K
                    1550
                    1560    ; si se llega aquí, la ROM del interfaz está conectada
C9B7    D7          1570    ROM2    RST 16                  ; las rutinas de la ROM de
C9B8    4D0D        1580            DEFW #D4D               ; 16K deben ser llamadas indirectamente.
C9BA    FDCB57C6    1590            SET 0, (IY+87)
C9BE    D7          1600            RST 16
```

```
C9BF   FDCB5786 1610   CONT1   RES 0, (IY+87)       ; quita el OVER 1
C9C3   C1       1620           POP BC               ; recupera la última posición de
C9C4   D1       1630           POP     DE;          dibujo y el contador
C9C5   15       1640           DEC     D
C9C6   20C2     1650           JR      NZ, BUCLE
                1660
C9C8   C1       1670           POP BC               ; recupera la posición inicial de
C9C9   78       1680           LD A, B              ; dibujo y mira si se ha alcanzado
C9CA   A7       1690           AND A                ; un borde.
C9CB   280D     1700           JR Z, PING           ; si es así ejecuta la acción
C9CD   FEAE     1710           CP 174               ; correspondiente
C9CF   2809     1720           JR Z, PING
C9D1   79       1730           LD A, C
C9D2   A7       1740           AND A
C9D3   2805     1750           JR Z, PING
C9D5   FEFE     1760           CP 254
C9D7   2801     1770           JR Z, PING
C9D9   C9       1780           RET                  ; sino es así, vuelve.
                1790
C9DA   C5       1800   PING    PUSH BC              ; salva las coordenadas del punto
C9DB   0600     1810           LD      B, O         ; para usar B como contador y
C9DD   78       1820   PINGL   LD      A, B         ; envía la salida al altavoz
                1830
                1840                                ; envía la salida al altavoz
C9DE   D3FE     1850           OUT (#FE), A         ; cambia el estado repetidamente
C9E0   10FB     1860           DJNZ PINGL           ; para conseguir un ruido
C9E2   C1       1870           POP BC
                1880
                1890   ; veamos de donde viene la llamada
                1900   ;
C9E3   2174C9   1910           LD HL, NRECU
C9E6   7D       1920           LD A, L
C9E7   E1       1930           POP HL
C9E8   E5       1940           PUSH HL
C9E9   AD       1950           XOR L
                1960
                1970   ; si no se ejecutara este retorno,
                1980   ; solo se pondrían los colores impares del borde
                1990   ;
                2000   ;
C9EA   C8       2010           RET Z
                2020
                2030   ; incrementa el color del borde (bits 0-2)
                2040   ;
C9EB   3AF5C9   2050           LD A, (COL)
C9EE   3C       2060           INCA
C9EF   32F5C9   2070           LD (COL), A
                2080   ;
                2090   ; y lo COLOCA
                2100   ;
C9F2   D3FE     2110           OUT (#FE), A
C9F4   C9       2120           RET
C9F5   00       2130   COL     DEFB 0
```

DeBASE

El programa DeBASE está formado por un conjunto de rutinas separadas que constituye un tosco, pero eficiente programa de base de datos, con rutinas para salvar y cargar en cinta y Microdrive.

El programa permitirá introducir datos, imprimir, buscar, borrar y modificar. No hay limitación en el número de registros ni en el tamaño de cada uno, excepto que deben caber en la pantalla. Se puede encontrar cualquier parte de cualquier ficha, y el cursor se colocará al principio de los datos que se estuviesen buscando. Una ficha puede cambiarse, ampliarse o reducirse después de su creación, sin destruir o alterar cualquier otra. Después de borrar una, todo el espacio usado por ella es liberado.

El programa usa sólo 1 byte de memoria por carácter, más 1 byte por ficha para marcar su final; se dispone de 32.000 bytes para escribir fichas y hay espacio para añadir otras utilidades.

```
                10      ; Esta subrutina imprimirá cualquier cosa que empiece
                20      ; con un byte con el bit 7 Alto y terminara en
                30      ; el siguiente byte con el bit 7 Alto.
                40      ; Tiene tres puntos de entrada y dos formas de uso. Si
                50      ; el registro A contiene 255 (#FF) en la entrada, entonces
                60      ; DE debe contener la dirección de inicio
                70      ; de un mensaje que
                80      ; debe estar precedido por un byte con el bit 7 alto
                90      ; para los otros valores de A se imprimirá el mensaje
                100     ; cuyo número se indica en A de una tabla de mensajes
                110     ; que empiece con un byte de marca
                120     ;
F424            130     MC1     ORG 62500
F424    F5      140     PMENSA  PUSH AF           ; esta entrada imprime en la
F425    D5      150             PUSH DE           ; parte superior izquierda de la pantalla
F426    3E02    160             LD A, 2
F428    CD0116  170             CALL #601         ; comprueba que está en la pantalla
                180                               ; principal, sino modificaría la
                190                               ; pantalla errónea, dejando la principal en su posición anterior
                200     ;
                210     ;
                220     ;
F42B    012118  230             LD BC, #1821
F42E    CDD90D  240             CALL 3545         ; usa la llamada a la ROM descrita en el
F431    D1      250             POP DE            ; capitulo 2
F432    F1      260             POP AF
F433    F5      270     IMPRIA  PUSH AF           ; esta entrada imprime en la posición
F434    D5      280             PUSH DE           ; actual de la pantalla principal.
F435    3E02    290             LD A, 2
F437    CD0116  300             CALL #1601
F43A    3EFF    310             LD A, 255         ; permite el scroll
F43C    328C5C  320             LD (23692), A     ; esta es la variable del
                330                               ; sistema SCR CT
F43F    3E0D    340             LD A, 13          ; ENTER (salta de línea a1
                350                               ; principio de la siguiente)
F441    D7      360             RST 16            ; el RESTART de impresión
F442    D1      370             POP DE
F443    F1      380             POP AF
F444    3C      390     IMPRI   INC A             ; esta entrada usa el canal actual
F445    1B      400             DEC DE            ; carga DE con la dirección de
                410                               ; inicio de la tabla
F446    2804    420             JR Z, IMPRIP      ; si A era #FF usa la dirección en DE
F448    3D      430             DEC A
F449    110000  440             LD DE, MENSA - 1  ; apunta al principio de los mensajes
F44C    CD0A0C  450     IMPRIP  CALL3082          ; rutina de impresión de la ROM
F44F    C9      460             RET
0001            470     MENSA   EQU
```

```
                10      ; Esta subrutina buscara en memoria una
                20      ; línea de caracteres que coincidan con
                30      ; los que empiezan en BCAR (Busca carácter) y que
                40      ; termine con la marca 255 (#FF)
                50      ; si hay alguna que coincida, se vuelve al principio
                60      ; de la ficha retrocediendo hasta que
                70      ; se encuentra un byte con el bit 7 alto. Entonces
                80      ; se la imprime con la rutina anterior.
                90      ;
                100     ; si el final de cada ficha es #8D entonces
                110     ; si el contenido de BCAR es #FF (no buscar nada) se
                120     ; encontrará el final de cada ficha
                130     ;
                140     ; A la salida el registro BC contendrá la dirección
                150     ; de inicio de la ficha o cero si no se ha hallado
                160     ;
                170     ;
F450            180     MC2     ORG #F450
F450    21CBF4  190     ENCUEN  LD HL, BCAR       ; la entrada principal
F453    7E      200             LD A, (HL)
F454    FEFF    210             CP #FF
F456    2006    220             JR NZ, BUSCA
F458    23      230             INC HL
F459    77      240             LD (HL), A
F45A    2B      250             DEC HL
```

```
F45B   3E8D      260          LD A, #8D
F45D   77        270          LD (HL), A
F45E   E5        280  BUSCA   PUSH HL
F45F   213275    290          LD HL, 30002
F462   01D084    300          LD BC, 34000
F465   D1        310          POP DE
F466   D5        320  MIRAMO  PUSH DE
F467   EDB1      330  MIRAEN  CPIR             ; mira el primer carácter
F469   2046      340          JR NZ, NOHAY     ; si no se encuentra
F46B   2B        350          DEC HL
F46C   22EEF4    360          LD (POSEN), HL   ; guarda la posición donde lo
F46F   ED43ECF4  370          LD (POSCON), BC  ; hallo y la cuenta de los que faltan.
F473   D1        380          POP DE
F474   D5        390          PUSH DE
F475   23        400  QUIZAS  INC HL           ; ahora se comprueba cada carácter
F476   EB        410          EX DE, HL
F477   23        420          INC HL
F478   7E        430          LD A, (HL)
F479   FEFF      440          CP 255           ; si se llega al final de la línea, entonces
F47B   280D      450          JR Z, HALLADO    ; la hemos hallado
F47D   EB        460          EX DE, HL
F47E   BE        470          CP (HL)
F47F   28F4      480          JR Z, QUIZAS
F481   2AEEF4    490          LD HL, (POSEN)   ; si la comprobación falla
                 500  ; busca desde el primer carácter
                 505  ; de nuevo.
F484   23        510          INC HL
F485   3ACBF4    520          LD A, (BCAR)
F488   18DD      530          JR MIRAEN
                 540  ;
F48A   E1        550  HALLAD  POP HL
F48B   2AEEF4    560          LD HL, (POSEN)   ; primero comprueba si hallado en
F48E   2B        570  ATRAS   DEC HL           ; el área permitida
F48F   22F0F4    580          LD (MEMPOS), HL
F492   23        590          INC HL
F493   E5        600          PUSH HL
F494   ED5BF2F4  610          LD DE, (LIMITE)
F498   A7        620          AND A
F499   ED52      630          SBC HL, DE
F49B   3014      640          JR NC, NOHAY     ; y si no, no le hace caso
F49D   El        650          POP HL
F49E   2B        660  ATRASM  DEC HL           ; antes de imprimir la entrada se
F49F   CB7E      670          BIT 7, (HL)      ; debe buscar el inicio
F4A1   28FB      675          JR Z, ATRASM
F4A3   23        680          INC HL
F4A4   E5        690          PUSH HL
F4A5   E5        700          PUSH HL
F4A6   CD6B0D    710          CALL 3435        ; borra toda la pantalla
F4A9   D1        720          POP DE
F4AA   3EFF      730          LD A, #FF        ; le indica a la rutina de impresión
F4AC   CD24F4    740          CALL PMENSA      ; que se use la dirección en DE
F4AF   C1        750          POP BC
F4B0   C9        760          RET
F4B1   3E03      770  NOHAY   LD A, 3          ; imprime mensaje 3
F4B3   CD33F4    780          CALL IMPRIA
F4B6   El        790          POP HL
F4B7   010000    800          LD BC, 0
F4BA   C9        810          RET
                 820  ;
F4BB   21CBF4    830  CONBUS  LD HL, BCAR      ; esta es la entrada para buscar
F4BE   E5        840          PUSH HL          ; otra aparición de la línea
F4BF   7E        850          LD A, (HL)
F4CO   ED4BECF4  860          LD BC, (POSCON)
F4C4   2AEEF4    870          LD HL, (POSEN)
F4C7   23        880          INC HL
F4C8   C367F4    890          JP MIRAEN
                 900  ;
                 910  ;
F4CB             920  BCAR    DEF 833
F4EC   0000      930  POSCON  DEFW 0           ; el contador después de una búsqueda
F4EE   3275      940  POSEN   DEFW 30002       ; la posición donde lo hemos hallado
F4F0   3175      950  MEMPOS  DEFW 30001       ; uno antes de la posición anterior
F4F2   C0F3      960  LIMITE  DEFW62400        ; el límite de la memoria que se usa
F424             970  PMENSA  EQU #F424        ; rutina de impresión, ver listado
F433             980  IMPRIA  EQU #F433        ; anterior en este apéndice
```

```
                  10      ; esta rutina borra una ficha cuando se usa en
                  20      ; unión de la anterior.
                  30      ;
                  40      ; a la salida el par de registros BC contendrá la
                  50      ; memoria que ha quedado libre.
                  60      ;
F4CB              70      MC3     ORG #F4CB
F4CB    CDCBF4    80      XFIC    CALL ENCUEN       ; se usa para encontrar
F4CE    78        90      XFICEN  LD A, B           ; la ficha a borrar.
F4CF    B1        100             OR        C
F4D0    C8        110             RET Z             ; BC=0 si no existe
                  111
F4D1    E5        120             PUSH HL
F4D2    C5        130             PUSH BC
F4D3    3E02      140             LD A, 2           ; mensaje 2
F4D5    CDD5F4    150             CALL IMPRIA
F4D8    3E7F      160     MTECLA  LD A, 127         ; este es un método de
                  170                               ; mirar si una tecla esta pulsada
F4DA    DBFE      180             IN A, (#FE)
F4DC    1F        190             RRA
F4DD    3030      200             JR NC, PARA
F4DF    3E7F      210             LD A, 127
F4E1    DBFE      220             IN A, (#FE)
F4E3    CB5F      230             BIT 3, A
F4E5    282E      240             JR Z, SIGFIC
F4E7    3EFE      250             LD A, 254
F4E9    DBFE      26O             IN A, (#FE)
F4EB    CB57      270             BIT 2, A
F4ED    20E9      280             JR NZ, MTECLA
F4EF    C1        290             POP BC
F4F0    E1        300             POP HL
F4F1    C5        310             PUSH BC
F4F2    C5        320             PUSH BC           ; BC= inicio de la ficha
F4F3    E1        330             POP HL
F4F4    010000    340             LD BC, 0
F4F7    CB7E      350     F_FIN   BIT 7, (HL)
F4F9    23        360             INC HL
F4FA    03        370             INC BC            ; el final de la ficha
F4FB    28FA      380             JR Z, F_FIN       ; debe ser hallado.
F4FD    D1        390             POP DE            ; ahora se mueve todo
F4FE    C5        400             PUSH BC           ; lo que está por
F4FF    E5        410             PUSH HL           ; encima hacia abajo
F500    C1        420             POP BC            ; para borrar la ficha
F501    E5        430             PUSH HL
F502    2A02F5    440             LD HL, (LIMITE)
F505    A7        450             AND A
F506    ED42      460             SBC HL, BC
F508    E5        470             PUSH HL
F509    C1        480             POP BC
F50A    E1        490             POP HL
F50B    EDBO      500             LDIR
F50D    C1        510             POP BC            ; la cantidad de espacio ganado.
F50E    C9        520             RET
F50F    E1        530     PARA    POP HL            ; limpia la pila y muestra no hallado
F510    C1        540             POP BC
F511    010000    550             LD BC, 0
F514    C9        560             RET
F515    El        570     SIGFIC  POP HL            ; recupera los detalles y sigue
F516    C1        580             POP BC            ; buscando.
F517    CD17F5    590             CALL CONBUS
F51A    C3CEF4    600             JP XFICEN
```

```
                  10      ; esta rutina permite guardar la entrada del teclado
                  20      ; en memoria a la vez que se saca a pantalla.
                  30      ;
                  40      ;
                  50      ; permite el movimiento izquierda/derecha del cursor así
                  60      ; como insertar y borrar caracteres.
                  70      ;
                  80      ; también ofrece una ayuda y un menú.
                  90      ;
                  100     ; como se muestra aquí es para usar con otras
```

```
                    110    ; rutinas de esta sección.
                    120    ; pero se puede modificar para otros fines.
                    130    ,
                    140    ; comprueba la memoria disponible y para la
                    150    ; entrada si se llega al límite.
                    160    ;
                    170    ; hay una rutina extra de entrada para búsqueda
                    180    ; así como rutinas de selección de opciones
                    190    ; esto se hace así para mostrar los diferentes
                    200    ; métodos de entrada y ver los problemas
                    210    ; mencionados en el texto.
                    220    ;
                    230    ;
                    240    ; esta es la dirección de entrada desde BASIC
                    250    ;
F51D                260    MC4     ORG #F51D
F51D    3EA0        270    USR     LD A, #A0           ; se comprueba la memoria libre
F51F    2A1FF5      280            LD HL, (LIMITE)
F522    ED5B22F5    290            LD DE, (CARPOS)
F526    A7          300            AND A
F527    ED52        310            SBC HL, DE
F529    D234F5      320            JP NC, NOLLEN
F52C    3E09        330            LD A, 9             ; mensaje 9
F52E    CD2EF5      340            CALL IMPRIA
F531    C3DFF6      350            JP MENS4
F534    E5          360    NOLLEN  PUSH HL             ; toda la memoria por encima
F535    C1          370            POP BC              ; de la última ficha hasta el limite
F536    D5          380            PUSH DE             ; es limpiada y marcada libre.
F537    E1          390            POP HL
F538    13          400            INC DE
F539    77          410            LD (HL), A
F53A    EDB0        420            LDIR
F53C    2A22F5      430    COGEA   LD HL, (CARPOS)     ; el inicio de la sig. ficha
F53F    2B          440            DEC HL
F540    7E          450            LD A, (HL)
F541    FE8D        460            CP #8D
F543    2801        470            JR Z, AINIC
F545    23          480            INC HL
F546    368D        490    AINIC   LD (HL), #8D
F548    2248F5      500            LD (CUPOS), HL      ; es marcado como inicio
F54B    CD6B0D      510            CALL 3435           ; la pantalla es borrada
F54E    CD4EF5      520            CALL ABRE1
F551    012118      530            LD BC, #1821        ; la posición de impresión de la
F554    CDD90D      540            CALL 3545           ; pantalla inferior es puesta arriba a
                    550                                ; la izquierda.
F557    3E07        560            LD A, 7             ; mensaje 7
F559    CD59F5      570            CALL IMPRI
F55C    3E01        580            LD A, l             ; el copyright
F55E    CD5EF5      590            CALL PMENSA
F561    AF          600            XOR A               ; la bandera del modo de edición es bajada.
F562    3262F5      610            LD (BANDE), A
F565    012116      620            LD BC, #1621        ; pone la línea 2, columna 0
F568    CDD90D      630            CALL 3545
F56B    3E3E        640            LD A, ">"
F56D    D7          650            RST 16
F56E    CD02F7      660            CALL CURSOR
F571    CD4EF5      670    ENTRA   CALL ABRE1
F574    CDA810      680    ENT1    CALL #10A8          ; esta es la rutina de entrada del
F577    3806        690            JR C, PULSAD        ; teclado, el acarreo puesto
                    700                                ; si se pulsa otra tecla
F579    FDCB029E    710            RES 3, (IY+2)       ; esta es TV FLAG, ver cap. 2
F57D    18F5        720            JR ENT1
F57F    FE08        730    PULSAD  CP 8                ; se comprueban las teclas de funciones
F581    284E        740            JR Z, ATRAS1
F583    FE09        750            CP 9
F585    CA11F7      760            JP Z, DER1
F588    FE0C        770            CP 12               ; CAPS SHIFT y 0
F58A    CA5CF7      780            JP Z, BORRA
F58D    FEC5        790            CP 197              ; OR
F58F    C8          800            RET Z
F590    FECD        810            CP 205              ; STEP
F592    CACFF7      820            JP Z, AYUDA
F595    FEC6        830            CP 198
F597    CA8EF7      840            JP Z, VALE_Y
F59A    FEC3        850            CP 195; NOT
F59C    CAD9F6      860            JP Z, MIRAM
```

```
F59F    FECC    870             CP 204          ; TO
F5A1    CAB7F7  880             JP Z, LIMPRE
F5A4    2A48F5  890             LD HL, (CUPOS)
F5A7    23      900             INC HL          ; ahora se mira si se ha llegado
F5A8    F5      910             PUSH AF         ; al final de una ficha y si es
F5A9    3A62F5  920             LD A, (BANDE)   ; así no se permite nada más.
F5AC    A7      930             AND A
F5AD    2806    940             JR Z, Y         ; si no se debe estar en edición
F5AF    7E      950             LD A, (HL)      ; memoria libre
F5B0    FE8D    960             CP #8D
F5B2    CA40F7  970             JP Z, FINFIC
F5B5    F1      980     VALE_Y  POP AF
F5B6    FEE2    990             CP 226          ; el STOP que para la entrada
F5B8    283B    1000            JR Z, PARA
F5BA    77      1010            LD (HL), A      ; el código del carácter es
F5BB    2248F5  1020            LD (CUPOS), HL  ; puesto en la posición de memoria
                1030                            ; actual y esta se
                1040                            ; actualiza para la siguiente.
F5BE    FE0D    1050            CP 13
F5C0    2006    1060            JR NZ, SACALE   ; si una entrada mueve el cursar
F5C2    CDC2F5  1070            CALL ABRE2
F5C5    CD02F7  1080            CALL CURSOR
F5C8    CDC2F5  1090    SACALE  CALL ABRE2
F5CB    D7      1100            RST 16          ; imprime la tecla y mueve el cursar
F5CC    CD02F7  1110            CALL CURSOR
F5CF    18A0    1120            JR ENTRA        ; va a por la siguiente tecla.
                1130    ;
                1140    ;
F5D1    2A48F5  1150    ATRAS1  LD HL, (CUPOS)  ; mueve el cursor
                1160                            ; atrás y borra
F5D4    7E      1170            LD A, (HL)      ; la memoria correspondiente.
F5D5    CB7F    1180            BIT 7, A
F5D7    2098    1190            JR NZ, ENTRA
F5D9    FE0D    1200            CP 13
F5DB    2B      1210            DEC HL
F5DC    2248F5  1220            LD (CUPOS), HL
F5DF    2005    1230            JR NZ, RETRO
F5E1    CDE1F5  1240            CALL ATRAS      ; pero ir atrás
                1250                            ; de un ENTER significa
F5E4    186F    1260            JR ENCRET       ; volver atrás y reimprimir.
F5E6    CDC2F5  1270    RETRO   CALL ABRE2
F5E9    CD02F7  1280            CALL CURSOR
F5EC    3E08    1290            LD A, 8
F5EE    D7      1300            RST 16
F5EF    CD02F7  1310            CALL CURSOR
F5F2    C371F5  1320            JP ENTRA        ; lee el siguiente carácter
                1330    ;
                1340    ;
F5F5    E5      1350    PARA    PUSH HL         ; si se pulsa STOP solo se marcar el
F5F6    2B      1360            DEC HL          ; final de la ficha si es nueva, pero
F5F7    7E      1370    FININ   LD A, (HL)      ; en todo caso se debe
                1380                            ; hallar el inicio
F5F8    23      1390            INC HL          ; de la memoria libre y actualizar el
F5F9    FEA0    1400            CP #A0          ; puntero.
F5FB    20FA    1410            JR NZ, FININ
F5FD    2B      1420            DEC HL
F5FE    2222F5  1430            LD (CARPOS), HL
F601    E1      1440            POP HL
F602    3A62F5  1450            LD A, (BANDE)
F605    A7      1460            AND A
F606    23      1470            INC HL
F607    2007    1480            JR NZ, PARAEN
F609    2222F5  1490            LD (CARPOS), HL
F60C    2B      1500            DEC HL
F60D    3E8D    1510            LD A, #8D
F60F    77      1520            LD (HL), A
F610    CD6B0D  1530    PARAEN  CALL 3435       ; se limpia la pantalla
F613    AF      1540            XOR A           ; pone normal la bandera del modo
F614    3262F5  1550            LD (BANDE), A
F617    3E04    1560            LD A, 4         ; se muestra el menú
F619    CD5EF5  1570            CALL PMENSA
F61C    CD4EF5  1580    ENMENU  CALL ABRE1
F61F    CDDE15  1590            CALL #15DE      ; la rutina de esperar entrada.
                1600                            ; ver capítulo 2
F622    FE0D    1610            CP 13
F624    CA1DF5  1620            JP Z, USR
```

```
F627    F620      1630             OR #20
F629    FE62      1640             CP "b"
F62B    2810      1650             JR Z, BENTRA
F62D    FE65      1660             CP "e"
F62F    281E      1670             JR Z, EENTRA
F631    FE63      1680             CP "c"
F633    CAD9F6    1690             JP Z, MIRAM
F636    FE73      1700             CP "s"
F638    CA38F6    1710             JP Z, SALCAR
F63B    18DF      1720             JR ENMENU
                  1730     ;
                  1740     ;
F63D    CDA0F6    1750     BENTRA  CALL ENCPT
F640    CD40F6    1760             CALL XFICH
F643    2A22F5    1770             LD HL, (CARPOS)   ; se recupera el espacio ganado
F646    A7        1780             AND A             ; moviendo el inicio
                  1790                               ; de la memoria libre
F647    ED42      1800             SBC HL, BC        ; hacia abajo todas
                  1810                               ; esas posiciones.
F649    2222F5    1820             LD (CARPOS), HL
F64C    C3DFF6    1830             JP MENS4
                  1840     ;
                  1850     ;
F64F    CDA0F6    1860     EENTRA  CALL ENCPT        ; empiece de la rutina
                  1870                               ; de encontrar
F652    CD52F6    1880             CALL ENCUEN
F655    78        1890     ENCRET  LD A, B           ; si no se halla el dato
                  1900                               ; vuelve al menú
F656    B1        1910             OR C
F657    CADFF6    1920             JP Z, MENS4
F65A    0B        1930             DEC BC
F65B    ED4348F51940               LD (CUPOS), BC    ; sino se pone en edición al
F65F    2A5FF6    1950             LD HL, (POSMEM)   ; principio de la ficha dada.
F662    A7        1960             AND A
F663    ED42      1970             SBC HL, BC
F665    2265F6    1980             LD (CURESP), HL
F668    3E01      1990             LD A, 1
F66A    3262F5    2000             LD (BANDE), A
F66D    CD4EF5    2010             CALL ABRE1
F670    012118    2020             LD BC, #1821
F673    CDD90D    2030             CALL 3545
F676    3E08      2040             LD A, 8
F678    CD59F5    2050             CALL IMPRI
F67B    3E07      2060             LD A, 7
F67D    CD59F5    2070             CALL IMPRI
F680    CDC2F5    2080             CALL ABRE2
F683    012118    2090             LD BC, #1821
F686    CDD90D    2100             CALL 3545
F689    3E0D      2110             LD A, 13
F68B    D7        2120             RST 16
F68C    CD02F7    2130             CALL CURSOR
F68F  ED4B65F6    2140             LD BC, (CURESP)
F693    78        2150     R1MAS   LD A, B
F694    B1        2160             OR C
F695    CA71F5    2170             JP Z, ENTRA
F698    C5        2180             PUSH BC
F699    CD17F7    2190             CALL DERCUR
F69C    C1        2200             POP BC
F69D    0B        2210             DEC BC
F69E    18F3      2220             JR R1MAS
                  2230     ;
                  2240     ;
F6A0    CD6B0D    2250     ENCPT   CALL 3435         ; es similar a la rutina de entrada
F6A3    3E05      2260             LD A, 5           ; anterior pero no permite
                  2270                               ; realizar ninguna
F6A5    CD5EF5    2280             CALL PMENSA       ; alteración y limita la entrada
F6A8    3E3E      2290             LD A, 62          ; cuando se alcanza LIM_EN
F6AA    D7        2300             RST 16
F6AB    21CBF6    2310             LD HL, BCAR+32
F6AE    22AEF6    2320             LD (LIM_EN), HL
F6B1    21ABF6    2330             LD HL, BCAR
                  2340     ;
                  2350     ;
F6B4    E5        2360     ENTRAF  PUSH HL           ; la rutina real de entrada
F6B5    CD4EF5    2370             CALL ABRE1
F6B8    CDDE15    2380             CALL #15DE
```

```
F6BB   E1       2390              POP HL
F6BC   FE0D     2400              CP 13              ; se termina al pulsar ENTER
F6BE   2816     2410              JR Z, PONFIN
F6C0   FE20     2420              CP 32
F6C2   38F0     2430              JR C, ENTRAF
F6C4   77       2440              LD (HL), A
F6C5   23       2450              INC HL
F6C6   E5       2460              PUSH HL
F6C7   CDC2F5   2470              CALL ABRE2
F6CA   D7       2480              RST 16
F6CB   E1       2490              POP HL
F6CC   EB       2500              EX DE, HL
F6CD   2AAEF6   2510              LD HL, (LIM_EN)
F6D0   A7       2520              AND A
F6D1   ED52     2530              SBC HL, DE
F6D3   EB       2540              EX DE, HL
F6D4   20DE     2550              JR NZ, ENTRAF
F6D6   36FF     2560     PONFIN   LD (HL), 255
F6D8   C9       2570              RET
                2580     ;
                2590     ;
F6D9   CDD9F6   2600     MIRAM    CALL CONBUS        ; busca otra aparición del texto
F6DC   C355F6   2610              JP ENCRET
                2620     ;
                2630     ;
F6DF   0603     2640     MENS4    LD B, 3            ; borra las tres líneas inferiores de
F6E1   CD440E   2650              CALL 3652          ; la pantalla como se dice en el cap. 2
F6E4   3E04     2660              LD A, 4            ; el menú
F6E6   CD2EF5   2670              CALL IMPRIA
F6E9   C31CF6   2680              JP ENMENU
                2690     ;
                2700     ;
                2710     ;
                2720     ; OVER 1 y OVER 0 ejecutan los mismos comandos que
                2730     ; el BASIC imprimiendo los códigos de control
F6EC   F5       2740     OVER1    PUSH AF
F6ED   E5       2750              PUSH HL
F6EE   3E15     2760              LD A, 21
F6F0   D7       2770              RST 16
F6F1   3E01     2780              LD A, 1
F6F3   D7       2790              RST 16
F6F4   El       2800              POP HL
F6F5   F1       2810              POP AF
F6F6   C9       2820              RET
F6F7   F5       2830     OVER0    PUSH AF
F6F8   E5       2840              PUSH HL
F6F9   3E15     2850              LD A, 21
F6F8   D7       2860              RST 16
F6FC   3E00     2870              LD A, 0
F6FE   D7       2880              RST 16
F6FF   E1       2890              POP HL
F700   F1       2900              POP AF
F701   C9       2910              RET
F702   F5       2920     CURSOR   PUSH AF            ; aquí se imprime el cursor y se
F703   CDECF6   2930              CALL OVER1         ; retrocede la posición de impresión
F706   3E5F     2940              LD A, 95           ; para colocarla encima de él.
F708   D7       2950              RST 16
F709   3EO8     2960              LD A, 8
F708   D7       2970              RST 16
F70C   CDF7F6   2980              CALL OVER0
F70F   F1       2990              POP AF
F710   C9       3000              RET
F711   CD17F7   3010     DER1     CALL DERCUR
F714   C371F5   3020              JP ENTRA
F717   CDC2F5   3030     DERCUR   CALL A8RE2         ; esta rutina mueve el cursor
F71A   2A48F5   3040              LD HL, (CUPOS)     ; a la derecha una
                3050                                 ; posición y cambia
F71D   23       3060              INC HL             ; el puntero de memoria al nuevo
                3070                                 ; carácter.
F71E   7E       3080              LD A, (HL)
F71F   C87F     3090              BIT 7, A
F721   CO       3100              RET NZ
F722   FE7F     3110              CP 127
F724   C8       3120              RET Z
F725   FE0D     3130              CP 13
F727   2803     3140              JR Z, ENRET
```

```
F729    FE20     3150            CP 32
F72B    D8       3160            RET C
F72C    2248F5   3170    ENRET   LD (CUPOS), HL
F72F    CD02F7   3180            CALL CURSOR
F732    FE0D     3190            CP 13
        F7342805 3200            JR Z, RETSI
F736    CDECF6   3210            CALL OVER1
        F7393E20 3220            LD A, 32
F738    D7       3230    RETSI   RST 16
F73C    CD02F7   3240            CALL CURSOR
F73F    C9       3250            RET
                 3260    ;
                 3270    ;
F740    E5       3280    FINFIC  PUSH HL          ; si se intentan introducir datos
F741    0603     3290            LD B, 3          ; después del final de
                 3300                             ; la ficha se produce
F743    CD440E   3310            CALL 3652        ; un aviso parpadeante y se borra 1a
F746    3E06     3320            LD A, 6          ; parte inferior de la pantalla.
F748    CD5EF5   3330            CALL PMENSA
F74B    FB       3340            EI               ; si estuviesen interrumpidas
F74C    0632     3350            LD B, 50         ; esta pausa de un segundo seria para
F74E    76       3360    ESPE1   HALT             ; siempre.
F74F    10FD     3370            DJNZ ESPE1
F751    E1       3380            POP HL
F752    F1       3390            POP AF
F753    2153F7   3400    REENT   LD HL, POSEN
F756    35       3410            DEC (HL)
F757    2B       3420            DEC HL
F758    34       3430            INC (HL)
F759    C3D9F6   3440            JP MIRAM
                 3450    ;
                 3460    ;
F75C    2A48F5   3470    BORRA   LD HL, (CUPOS)   ; funciona igual que
                 3480                             ; la de borrar
F75F    23       3490            INC HL           ; una ficha, pero con un carácter.
F760    7E       3500            LD A, (HL)
F761    CB7F     3510            BIT 7, A
F763    C271F5   3520            JP NZ, ENTRA
F766    E5       3530            PUSH HL
F767    E5       3540            PUSH HL
F768    2A1FF5   3550            LD HL, (LIMITE)
F76B    D1       3560            POP DE
F76C    A7       3570            AND A
F76D    ED52     3580            SBC HL, DE
F76F    E5       3590            PUSH HL
F770    C1       3600            POP BC
F771    D1       3610            POP DE
F772    DA71F5   3620            JP C, ENTRA
F775    CA71F5   3630            JP Z, ENTRA
F778    D5       3640            PUSH DE
F779    E1       3650            POP HL
F77A    23       3660            INC HL
F77B    EDB0     3670            LDIR
F77D    2A22F5   3680            LD HL, (CARPOS)
F780    28       3690            DEC HL
F781    2222F5   3700            LD (CARPOS), HL
F784    2A48F5   3710            LD HL, (CUPOS)
F787    23       3720            INC HL
F788    CDE1F5   3730            CALL ATRAS
F78B    C355F6   3740            JP ENCRET
                 3750    ;
                 3760    ;
F78E    2A48F5   3770    Y       LD HL, (CUPOS)   ; esta es la contraria.
F791    23       3780            INC HL
F792    E5       3790            PUSH HL
F793    2A1FF5   3800            LD HL, (LIMITE)
F796    D1       3810            POP DE
F797    A7       3820            AND A
F798    ED52     3830            SBC HL, DE
F79A    E5       3840            PUSH HL
F79B    C1       3850            POP BC
F79C    D5       3860            PUSH DE
F79D    ED5B1FF5 3870            LD DE, (LIMITE)
F7A1    D5       3880            PUSH DE
F7A2    E1       3890            POP HL
F7A3    2B       3900            DEC HL
```

```
F7A4    EDB8    3910            LDDR
F7A6    2A22F5  3920            LD HL, (CARPOS)
F7A9    23      3930            INC HL
F7AA    2222F5  3940            LD (CARPOS), HL
F7AD    E1      3950            POP HL
F7AE    3E20    3960            LD A, 32
F7B0    77      3970            LD (HL), A
F7B1    CDE1F5  3980            CALL ATRAS
F7B4    C355F6  3990            JP ENCRET
                4000    ;
                4010    ;
F7B7    2A48F5  4020    LIMPRE  LD HL, (CUPOS)      ; esta usa la corriente
                4030                                ; tres de un modo similar
F7BA    23      4040            INC HL              ; al canal 2 para la pantalla
F7BB    CDE1F5  4050            CALL ATRAS          ; si hay conectado un interfaz que
F7BE    C5      4060            PUSH BC             ; reconoce LPRINT, funcionara.
F7BF    3E03    4070            LD A, 3
F7C1    CD0116  4080            CALL #1601
F7C4    D1      4090            POP DE
F7C5    D5      4100            PUSH DE
F7C6    1B      4110            DEC DE
F7C7    AF      4120            XOR A
F7C8    CDC8F7  4130            CALL IMPRIP
F7CB    C1      4140            POP BC
F7CC    C355F6  4150            JP ENCRET
                4160    ;
                4170    ;
F7CF    CD6B0D  4180    AYUDA   CALL 3435
F7D2    3EOA    4190            LD A,10             ; la página de ayuda
F7D4    CD5EF5  4200            CALL PMENSA
F7D7    3EBF    4210    ESPERE  LD A, #BF
F7D9    DBFE    4220            IN A, (#FE)
F7DB    E601    4230            AND 1
F7DD    20F8    4240            JR NZ, ESPERE
F7DF    2A48F5  4250            LD HL, (CUPOS)
F7E2    2253F7  4260            LD (POSEN), HL
F7E5    23      4270            INC HL
F7E6    7E      4280            LD A, (HL)
F7E7    21ABF6  4290            LD HL, BCAR
F7EA    77      4300            LD (HL), A
F7EB    23      4310            INC HL
F7EC    36FF    4320            LD (HL), #FF
F7EE    C353F7  4330            JP REENT
```

```
                10      ; este es el inicio de las rutinas de salvar y cargar
                20      ; primero se calcula la longitud incluyendo un espacio
                30      ; para marcar el final y la longitud en bytes para
                40      ; reentrar después de cargar.
                50      ;
                60      ;
                70      ;
F7F1            80      MC5     ORG #F7F1
F7F1    2AF1F7  90      SALCAR  LD HL, (CARPOS)     ; primero se pone el final
F7F4    222E75  100             LD (29998), HL      ; de las fichas en el
                110                                 ; inicio del área
F7F7    112E75  120             LD DE, 29998        ; que se va a salvar o cargar.
F7FA    A7      130             AND A
F7FB    ED52    140             SBC HL, DE
F7FD    23      150             INC HL
F7FE    23      160             INC HL
F7FF    E5      170             PUSH HL
F800    3EOB    180             LD A, 11
F802    CD02F8  190             CALL PMENSA
F805    0619    200             LD B, 25            ; 25 interrupciones
F807    FB      210             EI
                220     ;
                230
                240     ; esto está mal hecho, pero es para demostrar el
                250     ; uso del HALT para retardar
                260     ;
                270     ;
F808    76      280     MANTEN  HALT                ; para asegurarse que no se coge
F809    10FD    290             DJNZ MANTEN         ; la S del menú principal
                300                                 ; ahora se explora el teclado
```

```
310     ;
F80B    3EBF     320     SAOLEE  LD A, #BF
F80D    DBFE     330             IN A, (#FE)
F80F    E602     340             AND 2
F811    CA11F8   350             JP Z, LEE
F814    3EFD     360             LD A, #FD
F816    DBFE     370             IN A, (#FE)
F818    E602     380             AND 2
F81A    20EF     390             JR NZ, SAOLEE
```

```
                 10      ; aquí se elige entre cinta y microdrive
                 20      ;
                 30      ;
F81C             40              ORG #F81C
F81C    C1       50      SALVA   POP BC
F81D    CD1DF8   60              CALL M_O_C        ; vuelve con el acarreo a 1 si se
                 70                                ; pulsa M y pone la cabecera
                 80      ;
F820    38FE     90              JR C, MSALVA
100     ;
110     ; esto salva una cabecera y un bloque de datos a
120     ; cinta. La cabecera es especial, ver salvando y
130     ; cargando en el capítulo 2
                 140     ;
F822    C5       150     CSALVA  PUSH BC           ; BC= longitud del bloque principal
                 160                               ; IX apunta ya al inicio
F823    110D00   170             LD DE, 13         ; la cabecera solo ocupa 13 bytes
F826    AF       180             XOR A             ; si A es cero indica cabecera
F827    CDC604   190             CALL 1222         ; rutina descrita en el cap. 2 que
                 200                               ; salva la cabecera.
F82A    D1       210             POP DE            ; recupera la longitud del bloque
                 220                               ; principal
F82B    DD212E75 230             LD IX, 29998      ; dirección de inicio del bloque
                 240                               ; principal de datos a salvar
F82F    3EFF     250             LD A, #FF         ; indica bloque principal
F831    CDC604   260             CALL 1222         ; usa la rutina de ROM de nuevo
F834    FB       270             EI                ; vuelve con las interrupciones
                 280                               ; desconectadas.
F835    C335F8   290             JP RETBA
```

```
                 10      ;
                 20      ; esto quita la ROM de 16 K
                 30      ; ver detalles en capítulo 3
                 40      ;
                 50      ;
F838             60              ORG #F838
F838    2149F8   70      CAMROM  LD HL,    ROMFUE
F83B    22ED5C   80              LD (23789), HL    ; la rutina de la ROM es accedida
                 90                                ; con el código de enlace 32 de
                 100                               ; modo que la pila se puede corromper.
F83E    D9       110             EXX               ; este es un método más seguro.
F83F    223FF8   120             LD (HLSAL), HL
F842    D9       130             EXX
F843    ED4343F8140              LD (BCSAL), BC
F847    CF       150             RST 8             ; usa el código de enlace
F848    32       160             DEFB #32
F849    E1       170     ROMFUE  POP HL            ; se deben quitar las dos
F84A    E1       180             POP HL            ; direcciones de retorno y
F84B    D1       190             POP DE            ; la de esta rutina
F84C    2A3D5C   200             LD HL, (23613)    ; esta es ERR_SP ver cap. 4
F84F    E5       210             PUSH HL           ; salva la posición del retorno
                 220                               ; de error en la pila para más
F850    2150F8   230             LD HL, MSLRET     ; tarde
F853    E5       240             PUSH HL           ; pone en la pila la nueva
F854    ED733D5C 250             LD (23613), SP    ; dirección de error
F858    D5       260             PUSH DE           ; y apunta SP para que
                 270                               ; recupere la dirección
F859    C9       280             RET               ; de retorno.
```

```
                 10      ; esta es la rutina de salvar a microdrive. ver cap. 3
                 20      ;
                 30      ;
```

```
F85A                40              ORG #F85A
F85A    CD5AF8      50      MSALVA  CALL CAMROM
F85D    CD5DF8      60              CALL MCABE          ; pone la cabecera
F860    CD60F8      70              CALL PONVAR         ; y las variables del sistema
F863    FDCB7CEE    80              SET 5, (IY+124)     ; indica que salva con FLAGS 3
F867    CD7F1E      90              CALL #1E7F          ; en realidad no se
                    100                                 ; necesita una llamada
                    110                                 ; ya que el retorno es por medio
                    120                                 ; del error 0 de la ROM de 16 K
F86A    El          130     MSLRET  POP HL              ; recupera la variable
                    140                                 ; del sistema ERR_SP
F86B    223D5C      150             LD (23613), HL      ; ERR_SP
F86E    FB          160             EI                  ; totalmente innecesario pero
                    170                                 ; seguro.
F86F    D9          180             EXX
F870    2A70F8      190             LD HL, (HLSAL)      ; recupera el registro HL'
F873    D9          200             EXX                 ; fíjese que la ROM de 16 K
                    210                                 ; esta activa ya que
F874    C374F8      220             JP LORET            ; el retorno es por medio de un
                    230                                 ; error del BASIC en la ROM de 16K
                    240
```

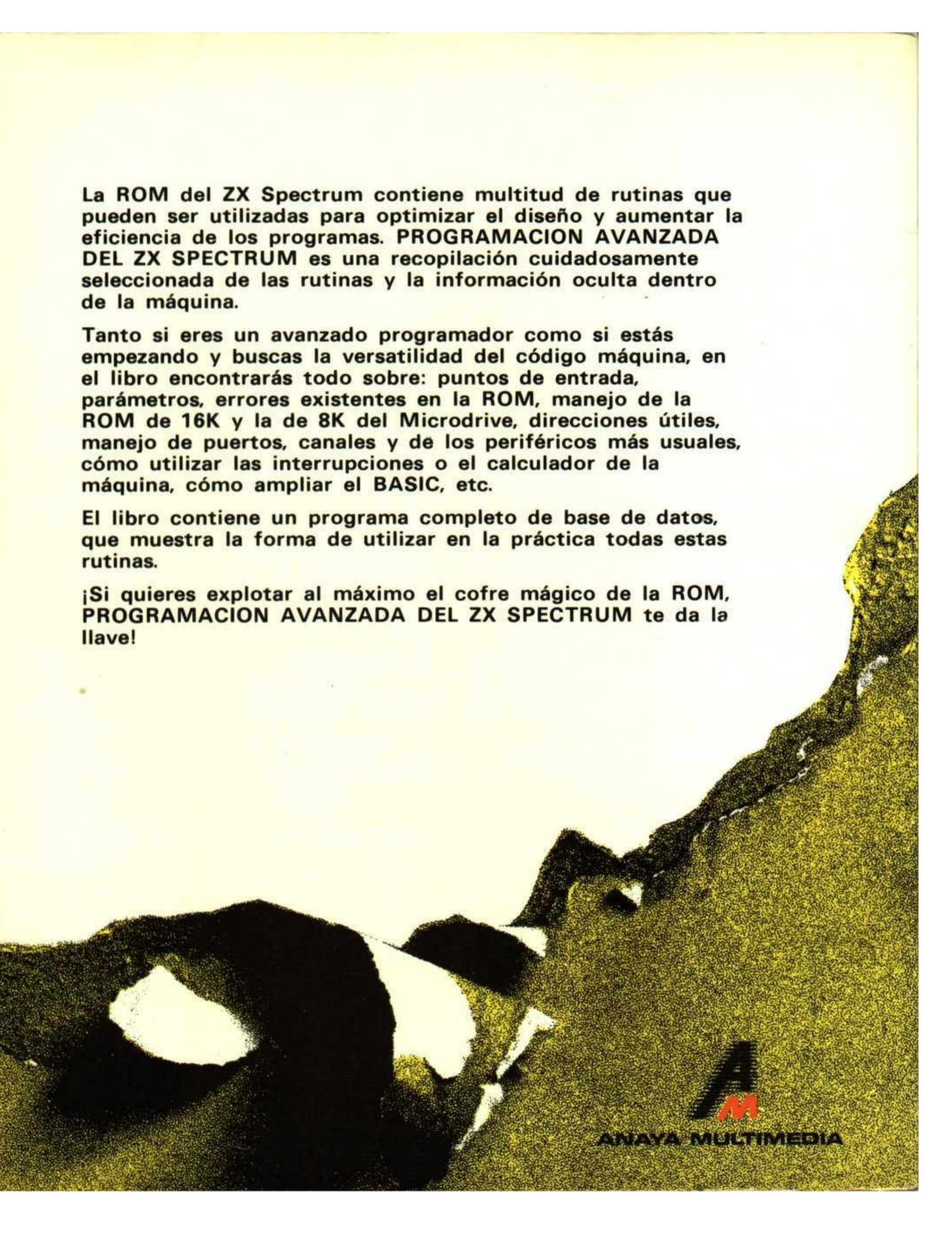
```
F877                10              ORG #F877
F877    21D95C      20      PONVAR  LD HL,23769         ; esto se explica en el cap. 3
F87A    364D        30              LD (HL), "M"
F87C    210AOO      40              LD HL, 10
F87F    22DA5C      50              LD (23770), HL
F8822   172F9       60              LD HL, BCAR+1
F8852   2DC5C       70              LD (23772), HL
F8882   10100       80              LD HL, 1
F88B    22D65C      90              LD (23766), HL
F88E    C9          100             RET
                    110     ;
                    120     ;
                    130     ;
                    140     ; esta es la rutina de leer del microdrive
                    150     ; todos los detalles se dan en el capitulo 3
                    160     ;
                    170     ;
F88F    010000      180     MLEE    LD BC, 0            ; los datos de la cabecera se tienen
F892    CD92F8      190             CALL CAMROM         ; que usar ya que no son conocidos
F895    CDA5F8      200             CALL MCABE
F898    CD77F8      210             CALL PONVAR
F89B    FDCB7CE6    220             SET 4, (IY+124)     ; esto indica leer
F89F    CDAF08      230             CALL #8AF           ; llama a la rutina de leer
F8A2    C3A2F8      240             JP MSLRET           ; en realidad es difícil llegar
                    250                                 ; aquí
                    260     ;
                    270     ;
                    280     ; esto pone la cabecera del microdrive
                    290     ; más detalles en el capítulo 3.
                    300     ;
                    310     ;
F8A521E65C          320     MCABE   LD HL,23782
F8A8    3603        330             LD (HL), 3
F8AA    23          340             INC HL
F8AB    ED5B5CF9    350             LD DE, (BCSAL)
F8AF    73          360             LD (HL), E
F8B0    23          370             INC HL
F8B1    72          380             LD (HL), D
F8B2    23          390             INC HL
F8B3    112E75      400             LD DE, 29998
F8B6    73          410             LD (HL), E
F8B7    23          420             INC HL
F8B8    72          430             LD (HL), D
F8B9    23          440             INC HL
F8BA    3680        450             LD (HL), #80
F8BC    C9          460             RET
                    470     ;
                    480     ;
                    490     ;
                    500     ;
                    510     ; decide leer de cinta o microdrive
                    520     ;
```

```
                ;
                ; lee de cinta como se indica en el cap. 2
                ;
F8BD   C1       LEE     POP BC
F8BE   CD05F9           CALL M_O_T
F8C1   38CC             JR C, MLEE
F8C3   AF       TLEE    XOR A            ; indica cabecera
F8C4   37               SCF              ; indica leer
F8C5   110D00           LD DE, 13
F8C8   14               INC D
F8C9   08               EX AF, AF'
F8CA   15               DEC D
F8C8   F3               DI
F8CC   DD2180F9         LD IX, BCAR+15
F8D0   CD6205           CALL #562
F8D3   FB               EI
F8D4   CD541F           CALL 8020
F8D7   D0               RET NC           ; BREAK pulsado
F8D8   DD2171F9 MIRCAB  LD IX, BCAR      ; se comprueba la cabecera
F8DC   2180F9           LD HL, BCAR+15
F8DF   060B             LD B, 11
F8E1   7E       MIRLP   LD A, (HL)
F8E2   DDBE00           CP (IX+OO)
F8E5   20DC             JR NZ, TLEE      ; equivocado. Cabecera mal
F8E7   DD23             INC IX           ; la cabecera está bien. Lee los datos.
F8E9   23               INC HL
F8EA   10F5             DJNZ MIRLP
F8EC   DD212E75         LD IX, 29998
F8F0   5E               LD E, (HL)
F8F1   23               INC HL
F8F2   56               LD D, (HL)
F8F3   37               SCF              ; indica lectura
F8F4   3EFF             LD A, #FF        ; bloque de datos principal
F8F6   14               INC D
F8F7   08               EX AF, AF'
F8F8   15               DEC D
F8F9   F3               DI
F8FA   CD6205           CALL #562
F8FD   FB               EI
F8FE   CD541F           CALL 8020
F901   D0               RET NC           ; BREAK pulsado
F902   C325F9           JP RETBA         ; cargado, vuelve a entrar
                ;
                ;
                ;
                ; esta rutina pide el nombre del fichero y si se
                ; lee de microdrive o de cinta
                ;
                ;
F905   CD2EF9   M_O_T   CALL SALVAT
F908   C5               PUSH BC
F909   DDE5             PUSH IX
F90B   3EOC             LD A, 12
F90D   CD0DF9           CALL IMPRIA
F910   3E7F     MIOCIN  LD A, #7F
F912   DBFE             IN A, (#FE)
F914   E604             AND 4
F916   37               SCF
F917   2808             JR Z, MOTSI
F919   3EFB             LD A, #FB
F91B   DBFE             IN A, (8FE)
F91D   E610             AND 16
F91F   20EF             JR NZ, MIOCIN
F921   DDE1     MOTSI   POP IX
F923   C1               POP BC
F924   C9               RET
                ;
                ;
                ;
                ; aquí se cogen los detalles de longitud del inicio
                ; del área leída o de lo que se puso ahí cuando
                ; se salvó.
                ;
F9252  A2E75    RETBA   LD HL, (29998)
F9282  60F9             LD (CARPOS), HL
F92B   C32BF9           JP USR
```

```
                  ;
                  ;
                  ;
                  ; esta rutina crea una cabecera en el área
                  ; BCAR de la memoria
                  ;
F92E   2171F9     SALVAT  LD HL, BCAR
F931   C5                 PUSH BC
F932   E5                 PUSH HL
F933   E5                 PUSH HL
F934   3603               LD (HL), 3
F936   C5                 PUSH BC
F937   060A               LD B, 10
F939   3E20               LD A, 32
F93B   23         PONCAB  INC HL
F93C   77                 LD (HL), A
F93D   10FC               DJNZ PONCAB
F93F   E5                 PUSH HL
F940   2264F9             LD (LIM_EN), HL
F943   E1                 POP HL
F944   C1                 POP BC
F945   23                 INC HL
F946   71                 LD (HL), C
F947   23                 INC HL
F948   70                 LD (HL), B
F949   CD6B0D             CALL 3435
F94C   3E0D               LD A, 13
F94E   CD4EF9             CALL PMENSA
F951   E1                 POP HL
F952   23                 INC HL
F953   CD53F9             CALL ENTRAF
F956   3620               LD (HL),32
F958   DDE1               POP IX
F95A   C1                 POP BC
F95B   C9                 RET
                  ;
                  ;
                  ;
F95C   0000       BCSAL   DEFW 0
F95E   0000       CUPOS   DEFW 0
F960   3175       CARPOS  DEFW 30001
F962   0000       HLSAL   DEFW 0
F964   0000       LIM_EN  DEFW 0
F966   30F2       LIMITE  DEFW 62000
F968   0000       POSCON  DEFW 0
F96A   0000       POSEN   DEFW 0
F96C   00         BANDE   DEFB 0
F96D   3175       POSMEM  DEFW 30001
F96F   2118       CURESP  DEFW #1821
F971              BCAR    DEFS 33
F992   80         MENSM1  DEFB #80
F993   80         MENS    DEFB #80
F994   44654241   MENS1   DEFM "DeBASE. ANAYA MULTIMEDIA 1985"
F9B1   8D                 DEFB #8D
F9B2   1001       MENS2   DEFW #110
F9B4   50554C53           DEFM "PULSE ESPACIO PARA PARAR"
F9CC   0D                 DEFB 13
F9CD   58205041           DEFM "X PARA BORRAR O N PARA LA SIGUIENTE FICHA"
F9F6   1000               DEFW #10
F9F8   8D                 DEFB #8D
F9F9   1002       MENS3   DEFW #210
F9FB   4E4F2O48           DEFM "NO HALLADO"
FA05   1000               DEFW #10
FA07   8D                 DEFB #8D
FA08   1001       MENS4   DEFW #110
FA0A   20202020           DEFM "    MENU"
FA19   0D0D               DEFW #0D0D
FA1B   50554C53           DEFM "PULSE:"
FA21   0D0D               DEFW #0D0D
FA23   45205041           DEFM "E PARA ENCONTRAR"
FA33   0D0D               DEFW #0D0D
FA35   42205041           DEFM "B PARA BORRAR"
FA42   0D0D               DEFW #0D0D
FA44   4320434F           DEFM "C CONTINUA LA BUSQUEDA"
FA5A   0D0D               DEFW #0D0D
FA5C   454E5445           DEFM "ENTER PARA METER OTRA FICHA"
```

```
FA77    0D0D     2050            DEFW #0D0D
FA79    53205041 206O            DEFM "S PARA SALVAR O LEER"
FA8D    1000     2070            DEFW #10
FA8F    8D       2080            DEFB #8D
FA90    5445434C 2090    MENS5   DEFM "TECLEE LOS DETALLES A         HALLAR"
FAAC    8D       2100            DEFB #8D
FAAD    1002     2110    MENS6   DEFW #210
FAAF    0D       2120            DEFB 13
FAB0    46494E20 2130            DEFM "FIN DE LA FICHA, NO MAS DATOS"
FACD    0D       2140            DEFB 13
FACE    41434550 2150            DEFM "ACEPTABLES"
FAD8    8D       2160            DEFB #8D
FAD9    1001     2170    MENS7   DEFW #110
FADB    50554C53 2180            DEFM "PULSE:"
FAE1    1002     2190            DEFW #210
FAE3    53544F50 2200            DEFM "STOP"
FAE7    1001     2210            DEFW #110
FAE9    50415241 2220            DEFM "PARA EL MENU,"
FAF6    1002     2230            DEFW #210
FAF8    02       2240            DEFB 2
FAF9    544F     2250            DEFM "TO"
FAFB    1001     2260            DEFW #110
FAFD    50415241 2270            DEFM "PARA IMPRIMIR O"
FB0C    1002     2280            DEFW #210
FB0E    53544550 2290            DEFM "STEP"
FB12    1001     2300            DEFW #110
FB14    50415241 2310            DEFM "PARA AYUDA"
FB1E    A0       2320            DEFB #A0
FB1F    1003     2330    MENS8   DEFW #310
FB21    45535441 2340            DEFM "ESTA EN MODO DE EDICION"
FB38    8D       2350    MC9     DEFB #8D
FB39    1002     2360    MENS9   DEFW #210
FB3B    53452051 2370            DEFM "SE QUEDO SIN MEMORIA"
FB4F    0D       2380            DEFB 13
FB50    53414C56 2390            DEFM "SALVE LAS FICHAS 0 SIMILAR"
FB6A    1000     2400            DEFW #l0
FB6C    8D       2410            DEFB #8D
FB6D    4355414E 2420    MENS10  DEFM "CUANDO ESTA EN MODO EDICION"
FB88    0D       2430            DEFB 13
FB89    50554C53 2440            DEFM "PULSANDO"
FB91    1002     2450            DEFW #210
FB93    4E4F54   2460            DEFM "NOT"
FB96    1000     2470            DEFW #010
FB98    454E434F 2480            DEFM "ENCONTRARA LA SIGUIENTE APARICION"
FBB9    4445204C 2490            DEFM "DE LA ULTIMA CADENA QUE SE BUSCO"
FBD9    0D0D     2500            DEFW #0D0D
FBDB    1002     2510            DEFW #210
FBDD    414E44   2520            DEFM "AND"
FBE0    1000     2530            DEFW #010
FBE2    494E5345 2540            DEFM "INSERTARA UN CARACTER"
FBF7    0D       2550            DEFB 13
FBF8    454E204C 2560            DEFM "EN LA POSICION DEL CURSOR"
FC11    0D0D     257O            DEFW #0D0D
FC13    44454C45 2580            DEFM "DELETE BORRARA EL CARACTER"
FC2D    0D       2590            DEFB 13
FC2E    454E204C 2600            DEFM "EN LA POSICION DEL CURSOR"
FC47    0D0D     2610            DEFW #0D0D
FC49    5349204E 2620            DEFM "SI NO HAY ESPACIO EN UNA FICHA"
FC67    0D       2630            DEFB 13
FC68    51554520 2640            DEFM "QUE ESTA MODIFICANDO, USE LA"
FC84    0D       2650            DEFB 13
FC85    46554E43 2660            DEFM "FUNCION INSERTAR PARA HACER SITIO"
FCA6    0D0D     2670            DEFW #0D0D
FCA8    4C415320 2680            DEFM "LAS TECLAS DE CURSOR PERMITEN"
FCC5    0D       2690            DEFB 13
FCC6    4D4F5645 2700            DEFM "MOVERSE POR EL TEXTO, PERO"
FCDF    0D       2710            DEFB 13
FCE0    4E4F2050 2720            DEFM "NO PUEDE IR ATRAS DE UN ENTER"
FCFD    0D0D     2730            DEFW #0D0D
FCFF    1004     2740            DEFW #410
FD01    50554C53 2750            DEFM "PULSE ENTER PARA VOLVER AL TEXTO'.
FD21    1000     2760            DEFW #10
FD23    8D       2770            DEFB #8D
FD24    50554C53 2780    MENS11  DEFM "PULSE S=SALVAR, L=LEER"
FD3A    8D       2790            DEFB #8D
FD3B    50554C53 2800    MENS12  DEFM "PULSE M PARA M/DRIVE"
```

```
FD4F    0D          2810            DEFB 13
FD50    4F205420    2820            DEFM "O T PARA CINTA"
FD5E    8D          2830            DEFB #8D
FD5F    5445434C    2840    MENS13  DEFM "TECLEE EL NOMBRE:"
FD70    8D          2850            DEFB #8D
FD71    8D          2860    ZFIN    DEFB #8D
```



La ROM del ZX Spectrum contiene multitud de rutinas que pueden ser utilizadas para optimizar el diseño y aumentar la eficiencia de los programas. **PROGRAMACION AVANZADA DEL ZX SPECTRUM** es una recopilación cuidadosamente seleccionada de las rutinas y la información oculta dentro de la máquina.

Tanto si eres un avanzado programador como si estás empezando y buscas la versatilidad del código máquina, en el libro encontrarás todo sobre: puntos de entrada, parámetros, errores existentes en la ROM, manejo de la ROM de 16K y la de 8K del Microdrive, direcciones útiles, manejo de puertos, canales y de los periféricos más usuales, cómo utilizar las interrupciones o el calculador de la máquina, cómo ampliar el BASIC, etc.

El libro contiene un programa completo de base de datos, que muestra la forma de utilizar en la práctica todas estas rutinas.

¡Si quieres explotar al máximo el cofre mágico de la ROM, **PROGRAMACION AVANZADA DEL ZX SPECTRUM** te da la llave!